Topic 12 Particle Filters: The art of hedging your bets

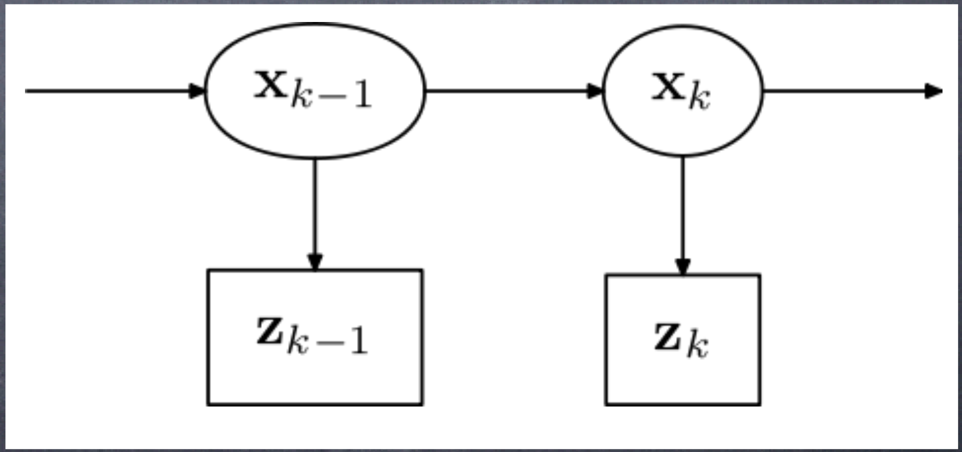
Bayesian filitering recap

posterior | likelihood | dynamics | prior

$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})\, d\mathbf{x}_{k-1}$$

"belief at t=k" | "update" | "predict" | "belief at t=k-1"

- distribution considers all possible robot poses
- assume one true pose
- recursive Markovian inference
- model, not algorithm



Bayesian filitering recap

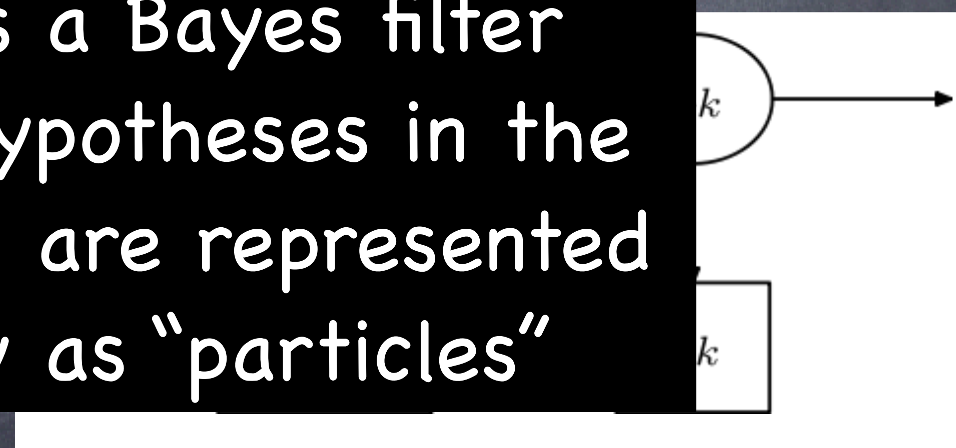
posterior	likelihood	dynamics	prior

$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})\, d\mathbf{x}_{k-1}$$

"belief at t=k"	"update"	"predict"	"belief at t=k-1"

- distribution considers all pos
- ass
- rec inference
- model, not algorithm

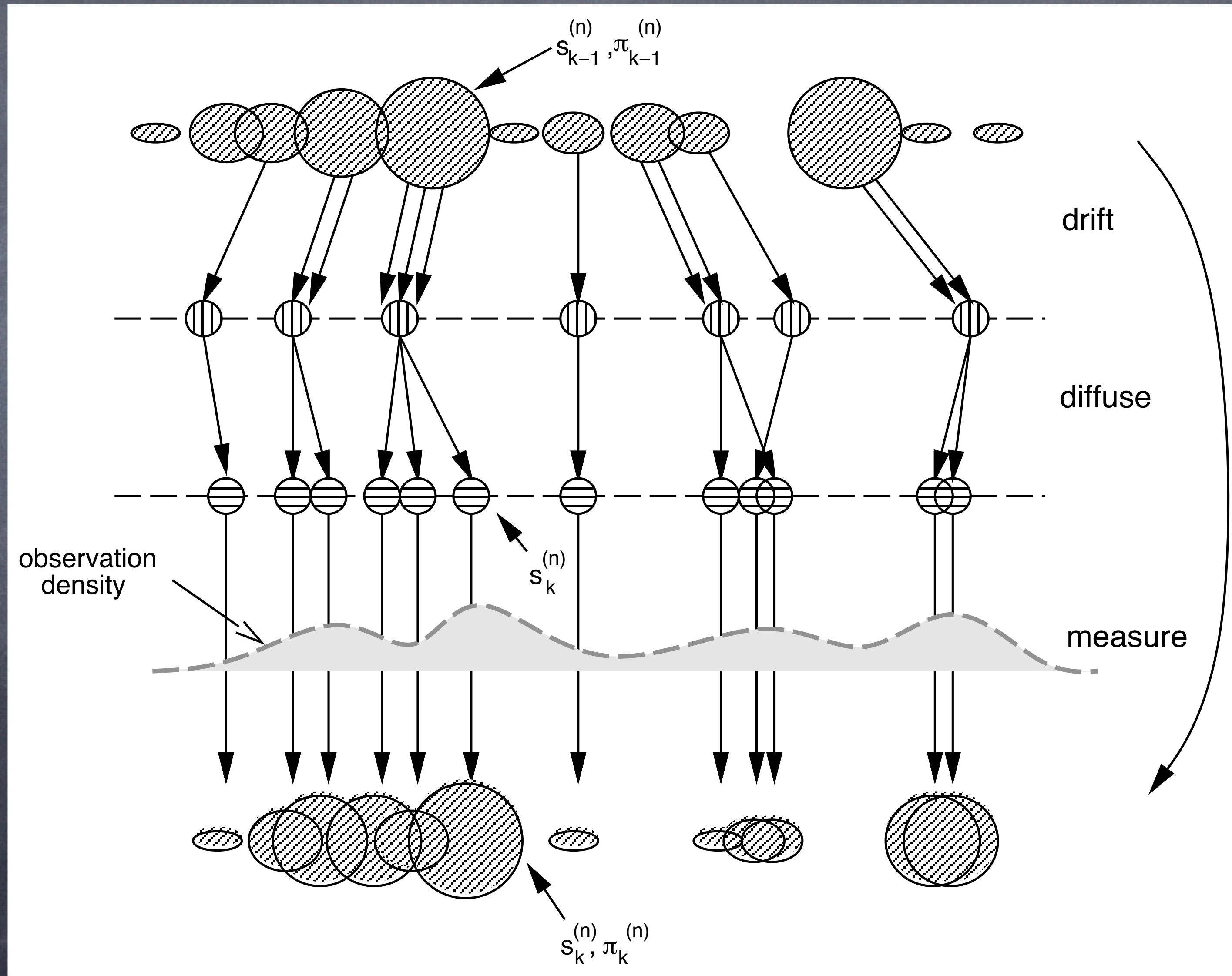
Particle filter is a Bayes filter where multiple hypotheses in the belief distribution are represented computationally as "particles"



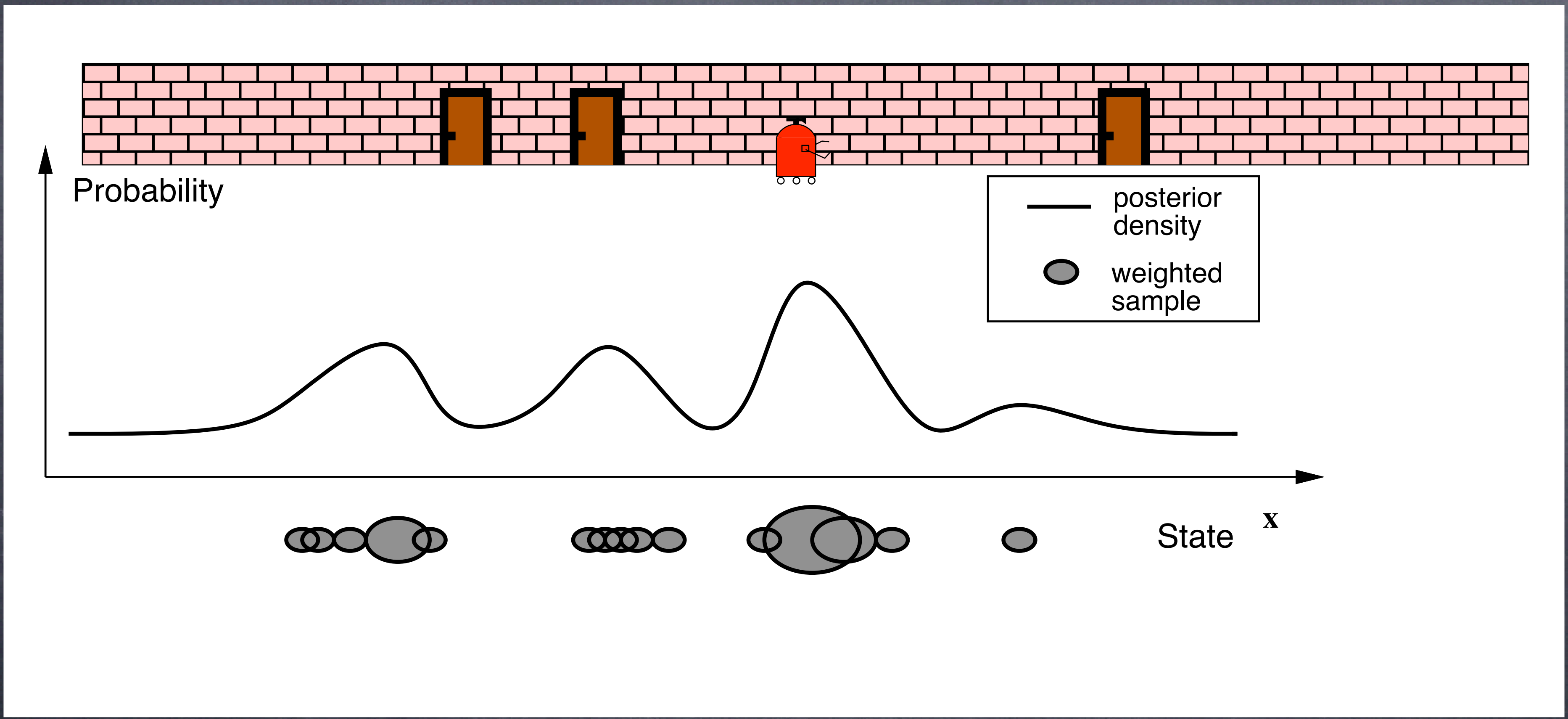
Condensation Algorithm

[Isard, Blake 1998]

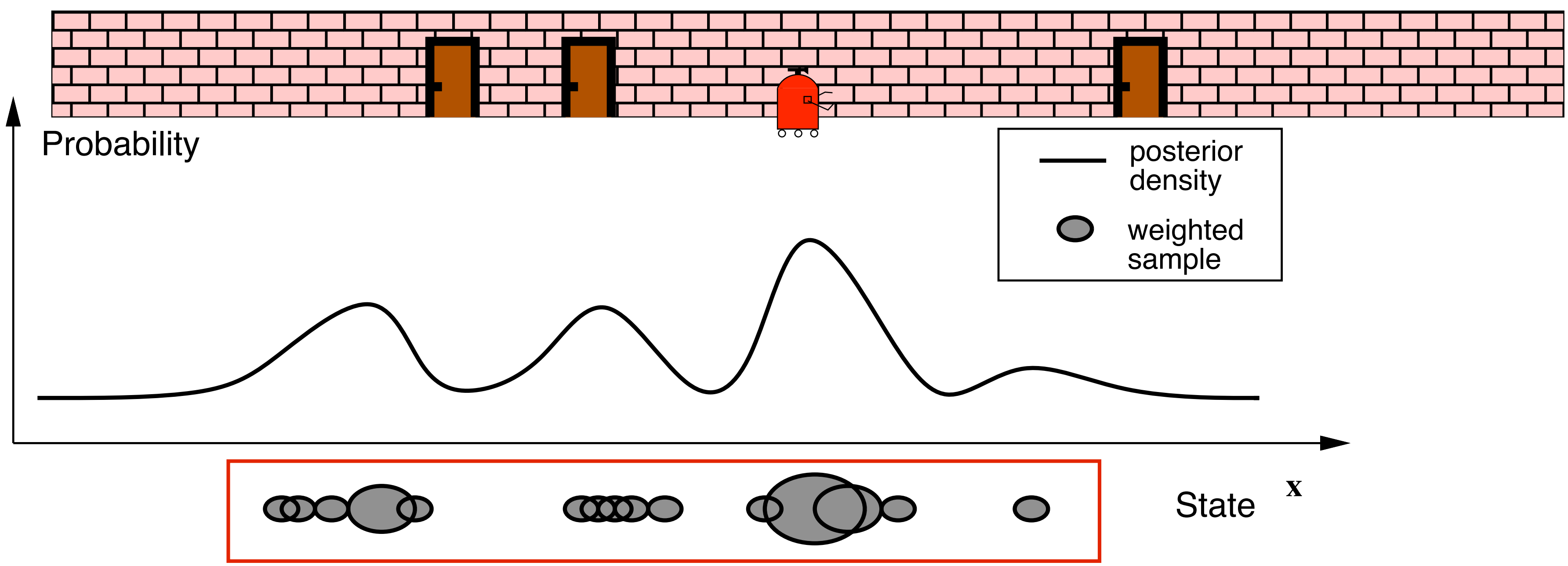
- Condensation is one algorithm for particle filtering
- State belief represented as particle hypotheses



Representing belief as particles



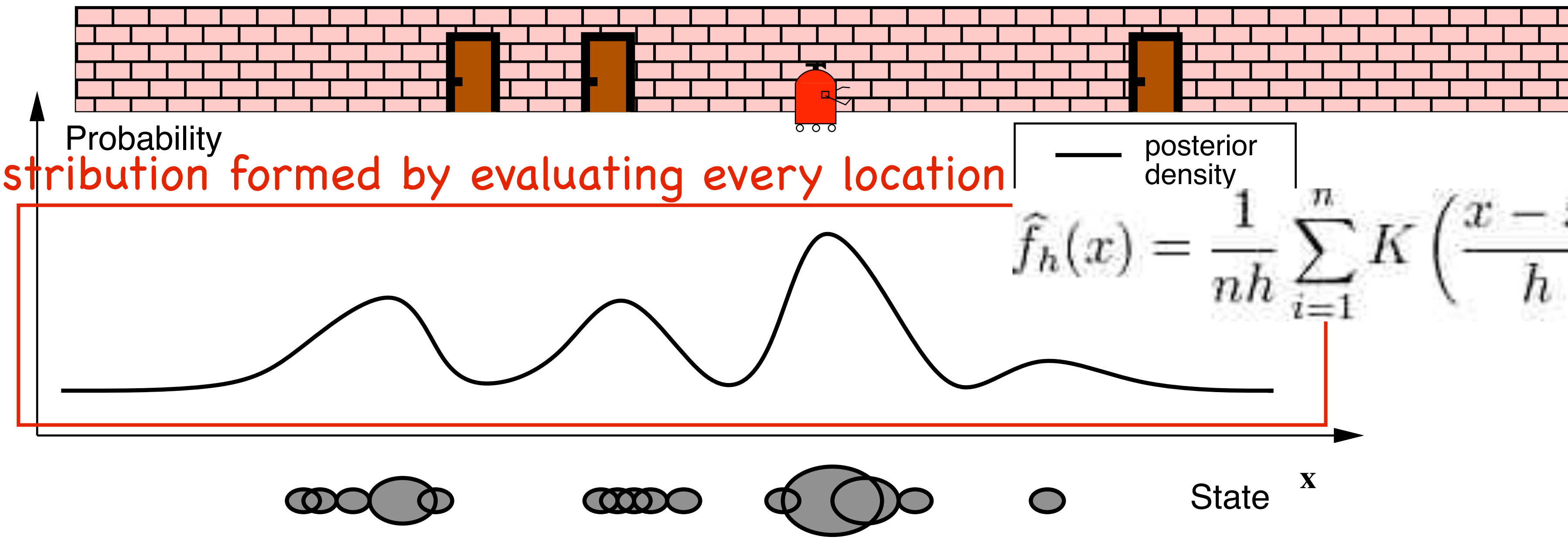
Representing belief as particles



each particle is a hypothesis on the robot's pose, weighted by the probability of the pose

Representing belief as particles

distribution formed by evaluating every location



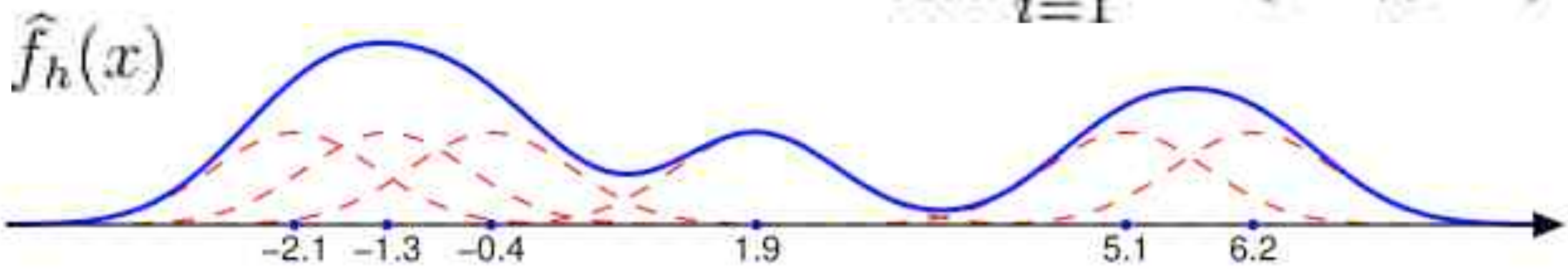
$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

each particle is a hypothesis on the robot's pose, weighted by the probability of the pose

Representing belief as particles

Tangent: How do particles form a distribution?

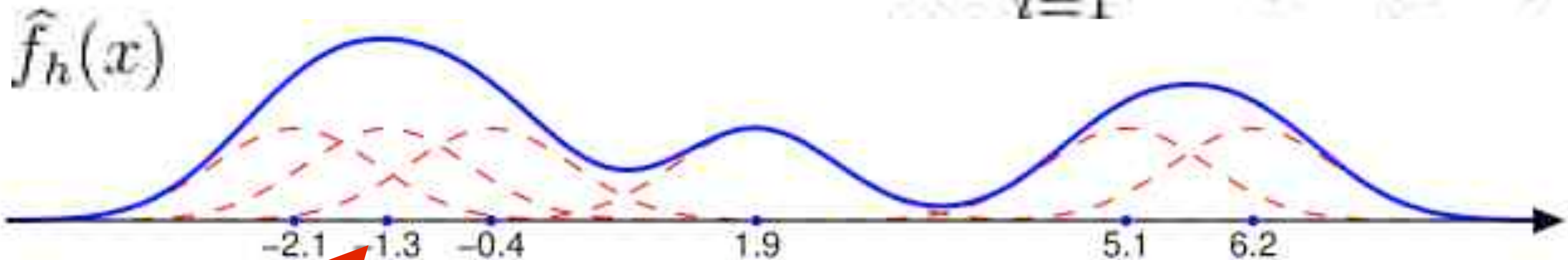
$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$



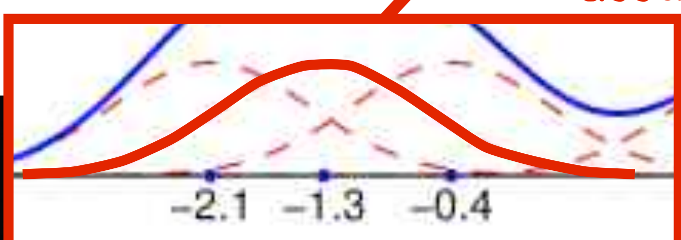
Representing belief as particles

Tangent: How do particles form a distribution?

$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$



assume a “kernel” function K at every point,
a “Gaussian” in this case



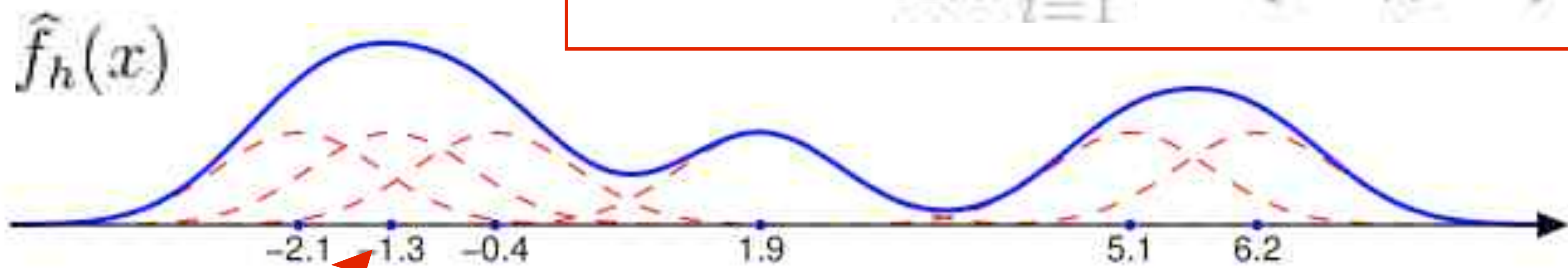
$$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

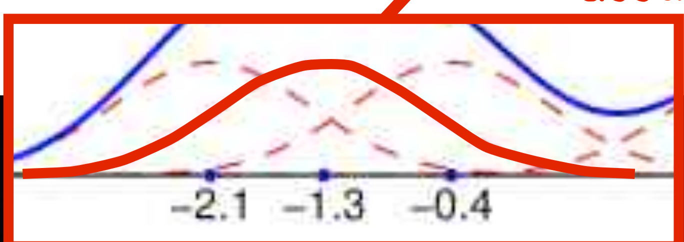
Representing belief as particles

Tangent: How do particles form a distribution?

kernel shape results from evaluating one K across all x

$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$





assume a “kernel” function K at every point,
a “Gaussian” in this case

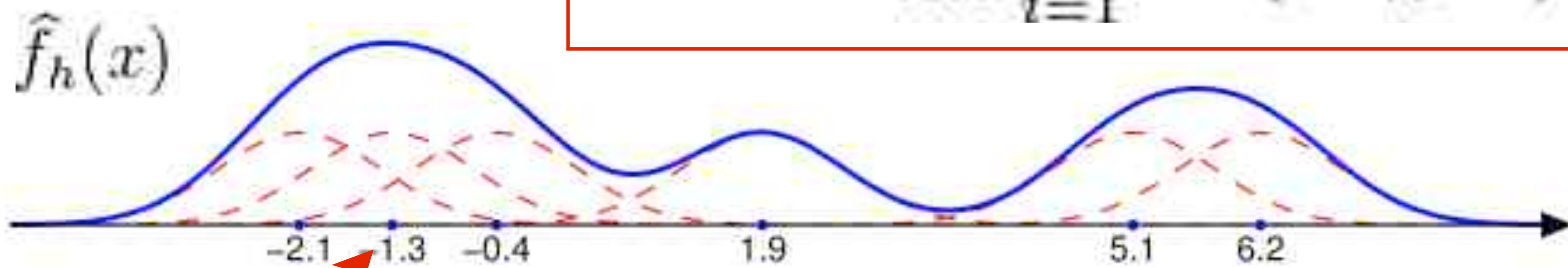
$$\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Representing belief as particles

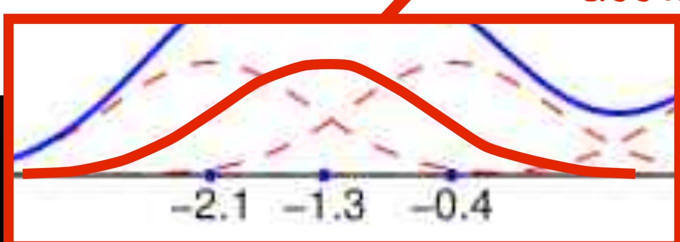
Tangent: How do particles form a distribution?

distribution results from evaluating all kernels K across all x

$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

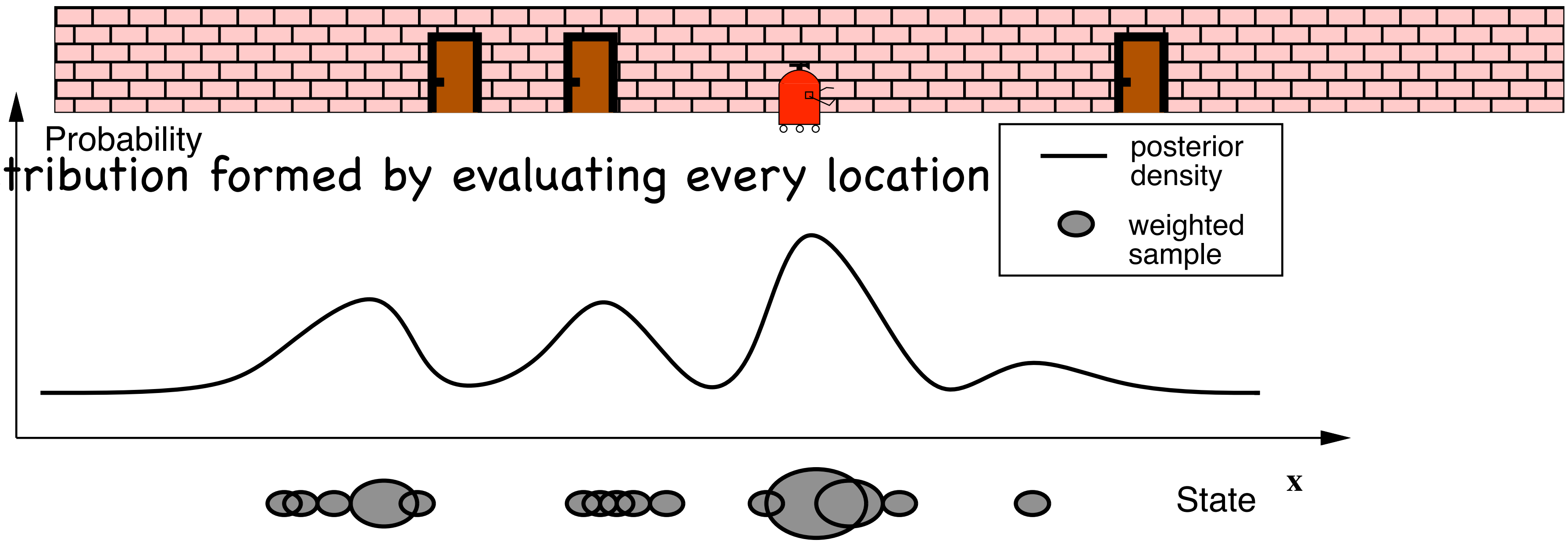


assume a "kernel" function K at every point,
a "Gaussian" in this case



$$\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

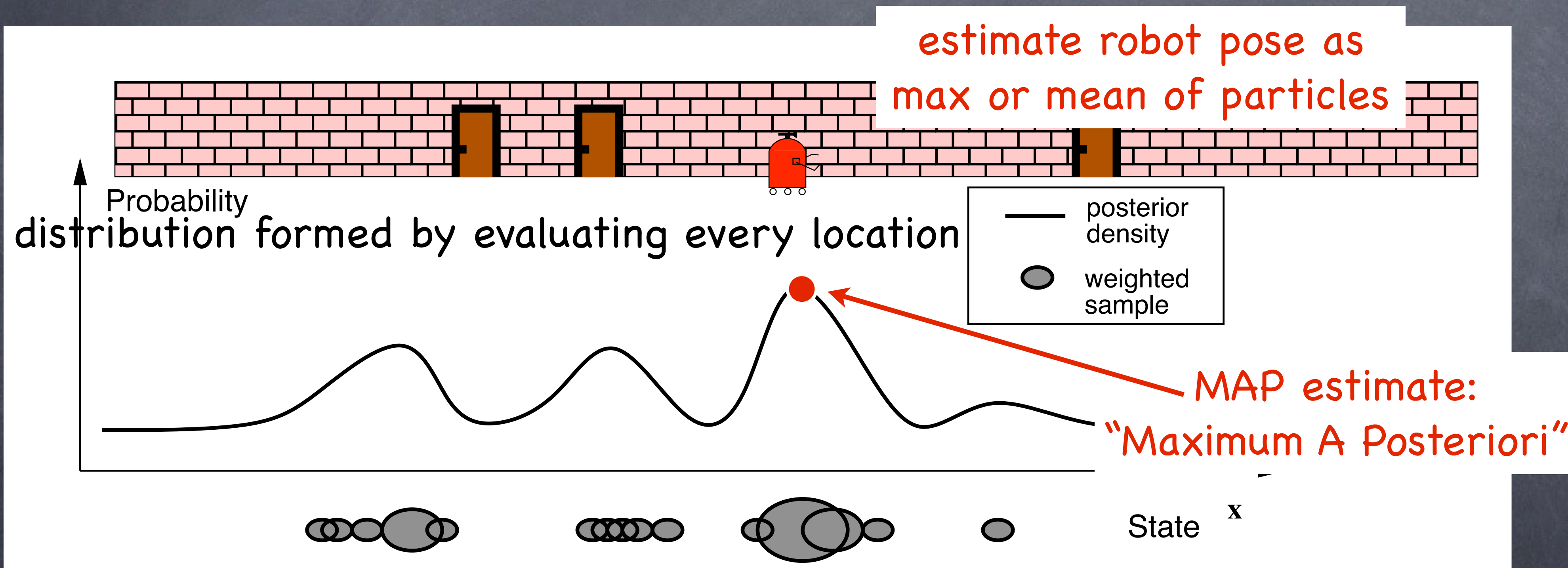
Representing belief as particles



distribution formed by evaluating every location

each particle is a hypothesis on the robot's pose, weighted by the probability of the pose

Representing belief as particles

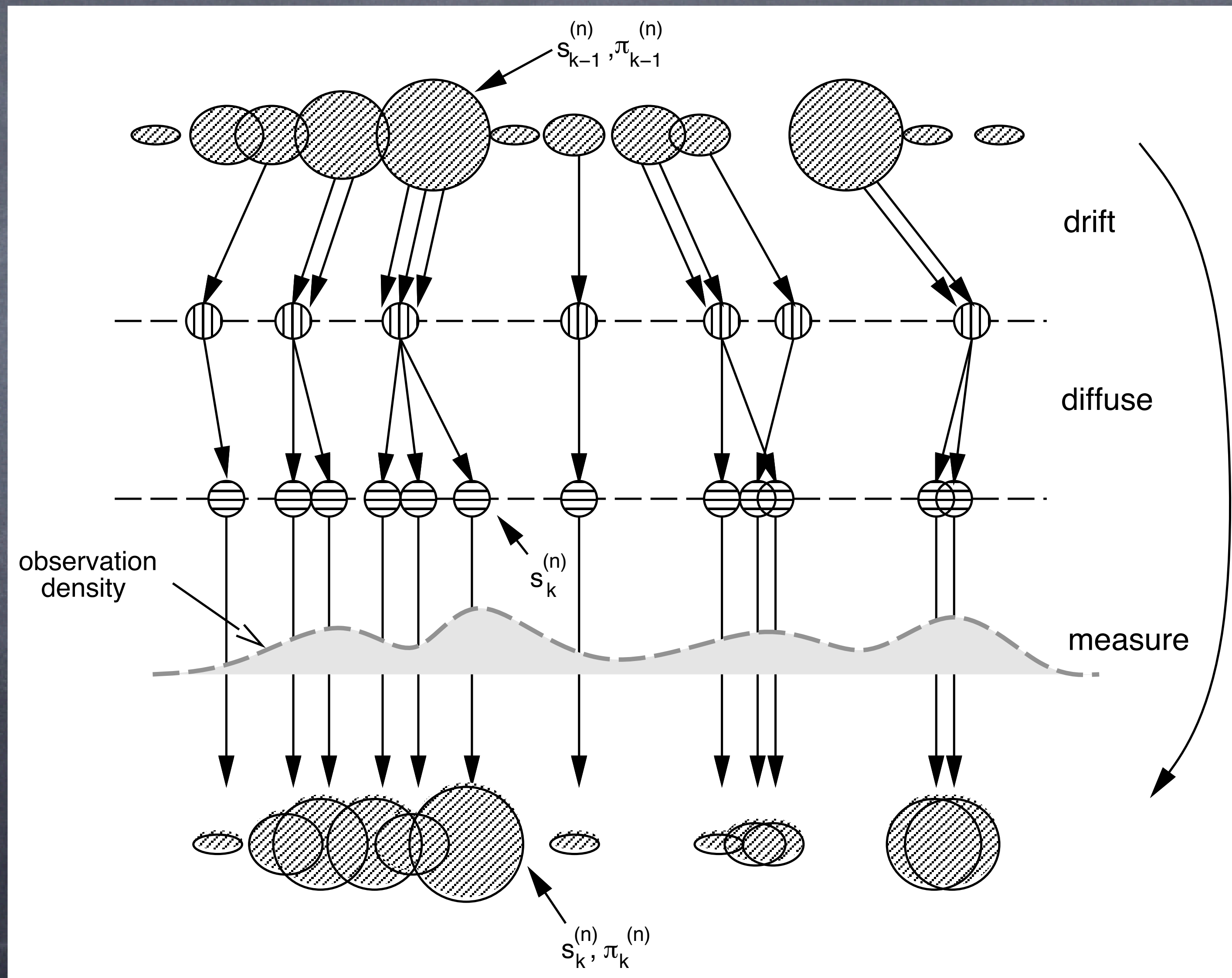


each particle is a hypothesis on the robot's pose, weighted by the probability of the pose

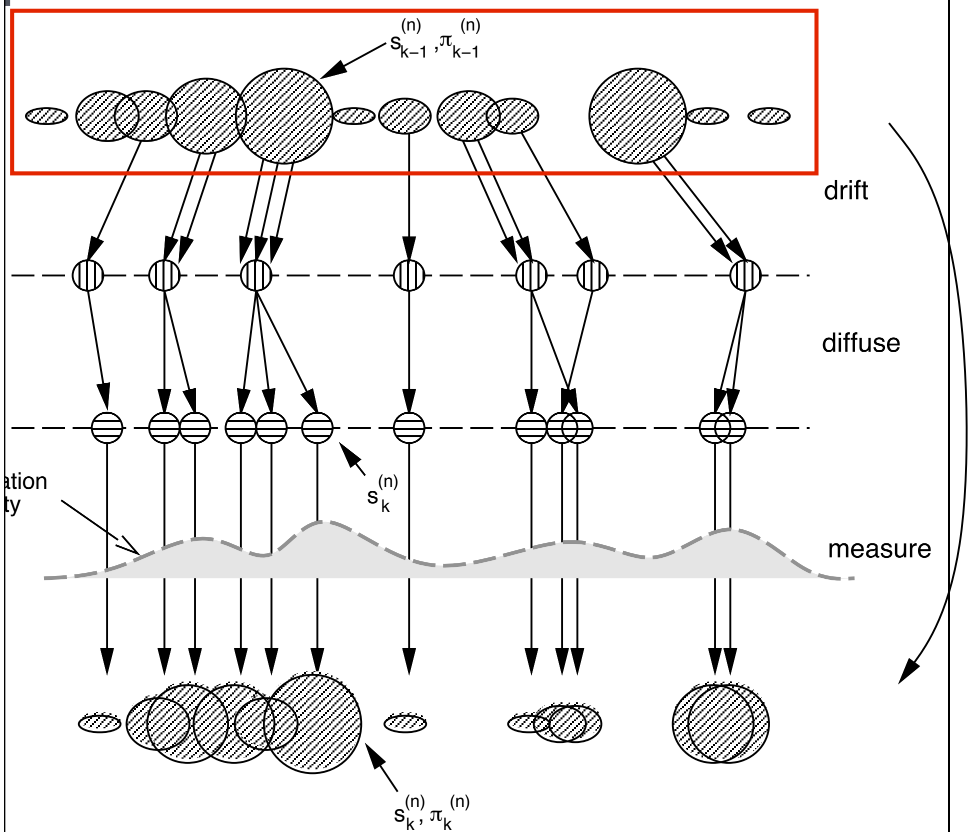
Condensation Algorithm

[Isard, Blake 1998]

- Condensation is one algorithm for particle filtering



sample set:
particle locations and weights



Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \dots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \dots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s}_t'^{(n)}$ as follows:
   (a) generate a random number $r \in [0, 1]$, uniformly distributed.
   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$
   (c) set $\mathbf{s}_t'^{(n)} = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from

$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s}_t'^{(n)})$$

   to choose each $\mathbf{s}_t^{(n)}$. For instance, in the case that the dynamics are governed by a linear stochastic differential equation, the new sample value may be generated as: $\mathbf{s}_t^{(n)} = \mathbf{A}\mathbf{s}_t'^{(n)} + B\mathbf{w}_t^{(n)}$ where $\mathbf{w}_t^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance — see section 5.
3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:

$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$

   then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where

$$\begin{aligned} c_t^{(0)} &= 0, \\ c_t^{(n)} &= c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1, \dots, N). \end{aligned}$$

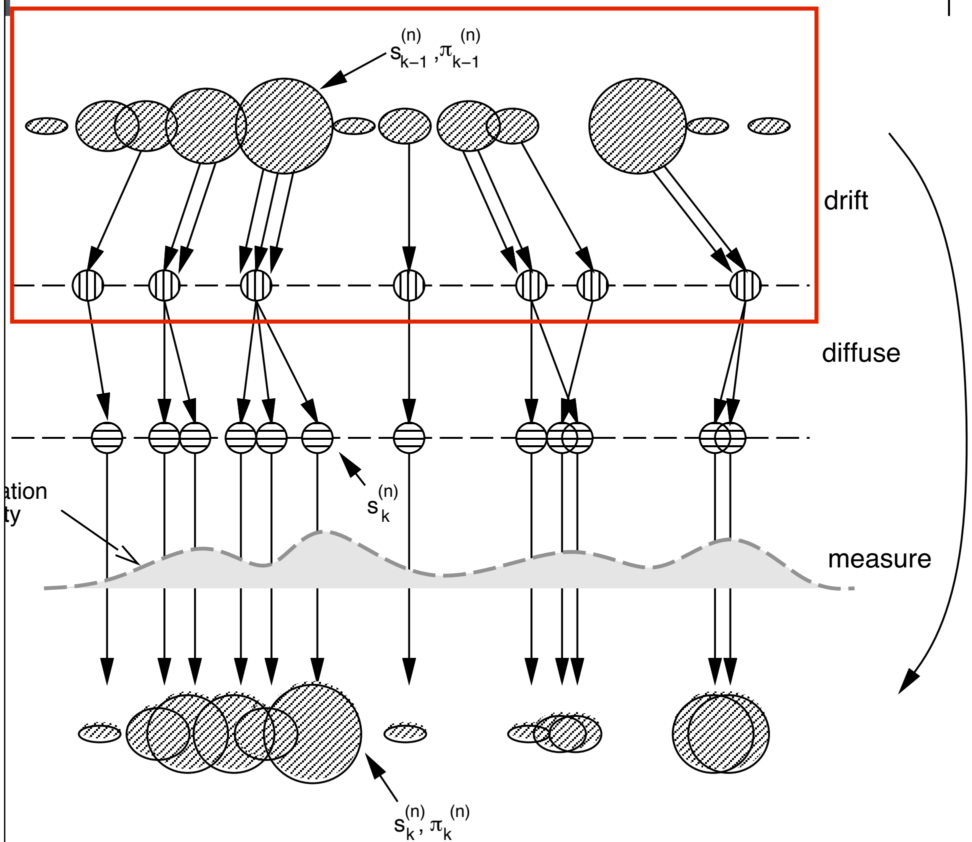
Once the N samples have been constructed: **estimate**, if desired, moments of the tracked position at time-step t as

$$\mathcal{E}[f(\mathbf{x}_t)] = \sum_{n=1}^{N} \pi_t^{(n)} f\left(\mathbf{s}_t^{(n)}\right)$$

obtaining, for instance, a mean position using $f(\mathbf{x}) = \mathbf{x}$.

Figure 6: **The Condensation algorithm.**

create new particle set based on "importance" of current particles



Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \ldots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s'}_t^{(n)}$ as follows:
   (a) generate a random number $r \in [0, 1]$, uniformly distributed.
   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$
   (c) set $\mathbf{s'}_t^{(n)} = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from

$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s'}_t^{(n)})$$

   to choose each $\mathbf{s}_t^{(n)}$. For instance, in the case that the dynamics are governed by a linear stochastic differential equation, the new sample value may be generated as: $\mathbf{s}_t^{(n)} = \mathbf{A}\mathbf{s'}_t^{(n)} + B\mathbf{w}_t^{(n)}$ where $\mathbf{w}_t^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance — see section 5.
3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:

$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$

   then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where

$$\begin{aligned} c_t^{(0)} &= 0, \\ c_t^{(n)} &= c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1, \ldots, N). \end{aligned}$$

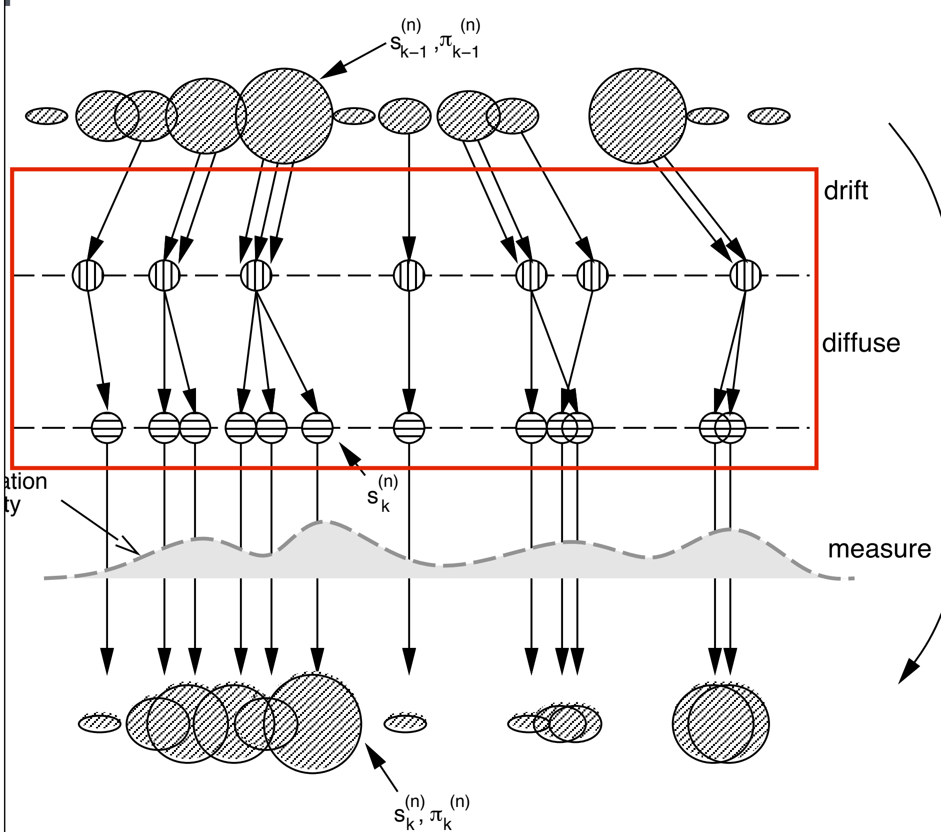
Once the N samples have been constructed: **estimate**, if desired, moments of the tracked position at time-step t as

$$\mathcal{E}[f(\mathbf{x}_t)] = \sum_{n=1}^{N} \pi_t^{(n)} f\left(\mathbf{s}_t^{(n)}\right)$$

obtaining, for instance, a mean position using $f(\mathbf{x}) = \mathbf{x}$.

Figure 6: **The Condensation algorithm**

predict movement of particles based on dynamics with noise



Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \ldots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s'}_t^{(n)}$ as follows:
   (a) generate a random number $r \in [0, 1]$, uniformly distributed.
   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$
   (c) set $\mathbf{s'}_t^{(n)} = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from

$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s'}_t^{(n)})$$

   to choose each $\mathbf{s}_t^{(n)}$. For instance, in the case that the dynamics are governed by a linear stochastic differential equation, the new sample value may be generated as: $\mathbf{s}_t^{(n)} = A\mathbf{s'}_t^{(n)} + B\mathbf{w}_t^{(n)}$ where $\mathbf{w}_t^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance — see section 5.
3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:

$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$

   then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where

$$\begin{aligned} c_t^{(0)} &= 0, \\ c_t^{(n)} &= c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1, \ldots, N). \end{aligned}$$

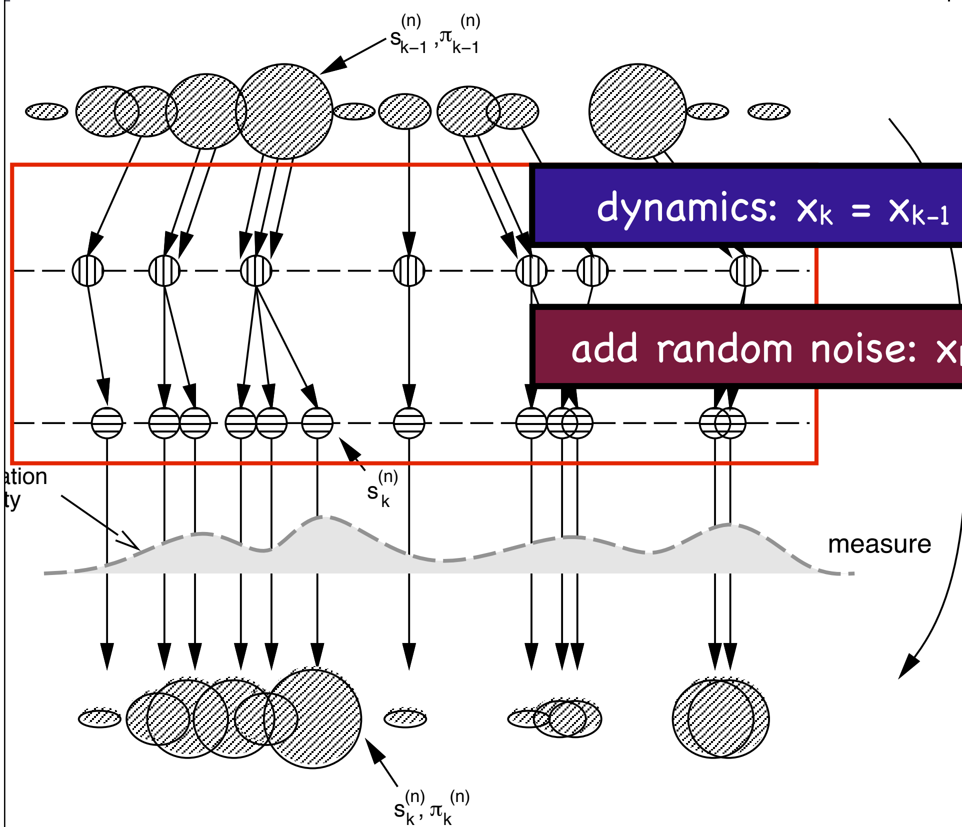
Once the N samples have been constructed: **estimate**, if desired, moments of the tracked position at time-step t as

$$\mathcal{E}[f(\mathbf{x}_t)] = \sum_{n=1}^{N} \pi_t^{(n)} f\left(\mathbf{s}_t^{(n)}\right)$$

obtaining, for instance, a mean position using $f(\mathbf{x}) = \mathbf{x}$.

Figure 6: **The Condensation algorithm**

predict movement of particles based on dynamics with noise



dynamics: $x_k = x_{k-1} + f(x_{k-1}, x_{k-1})$

add random noise: $x_k = x_k + \varepsilon$

Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \ldots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s}_t'^{(n)}$ as follows:

   (a) generate a random number $r \in [0, 1]$, uniformly distributed.

   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$

   $\mathbf{s}_{t-1}^{(j)}$

2. **Predict** by sampling from

$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s}_t'^{(n)})$$

to choose each $\mathbf{s}_t^{(n)}$. For instance, in the case that the dynamics are governed by a linear stochastic differential equation, the new sample value may be generated as: $\mathbf{s}_t^{(n)} = \mathbf{A}\mathbf{s}_t'^{(n)} + B\mathbf{w}_t^{(n)}$ where $\mathbf{w}_t^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance — see section 5.

3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:

$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$

then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where

$$c_t^{(0)} = 0,$$
$$c_t^{(n)} = c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1, \ldots, N).$$

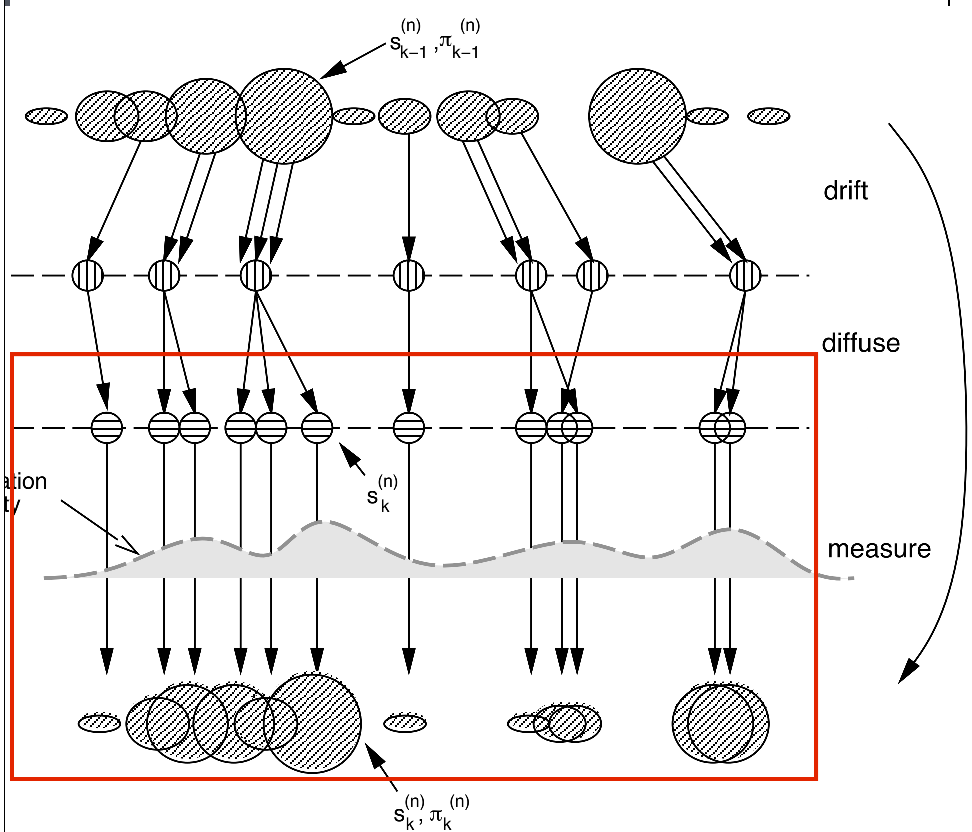
Once the N samples have been constructed: **estimate**, if desired, moments of the tracked position at time-step t as

$$\mathcal{E}[f(\mathbf{x}_t)] = \sum_{n=1}^{N} \pi_t^{(n)} f\left(\mathbf{s}_t^{(n)}\right)$$

obtaining, for instance, a mean position using $f(\mathbf{x}) = \mathbf{x}$.

Figure 6: **The** CONDENSATION **algorithm.**

update particle weights based on observed likelihood



Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \ldots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s}_t'^{(n)}$ as follows:
   (a) generate a random number $r \in [0, 1]$, uniformly distributed.
   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$
   (c) set $\mathbf{s}_t'^{(n)} = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from

$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s}_t'^{(n)})$$

   to choose each $\mathbf{s}_t^{(n)}$. For instance, in the case that the dynamics are governed by a linear stochastic differential equation, the new sample value may be generated as: $\mathbf{s}_t^{(n)} = \mathbf{A}\mathbf{s}_t'^{(n)} + B\mathbf{w}_t^{(n)}$ where $\mathbf{w}_t^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance — see section 5.
3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:

$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$

   then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where

$$\begin{aligned} c_t^{(0)} &= 0, \\ c_t^{(n)} &= c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1, \ldots, N). \end{aligned}$$

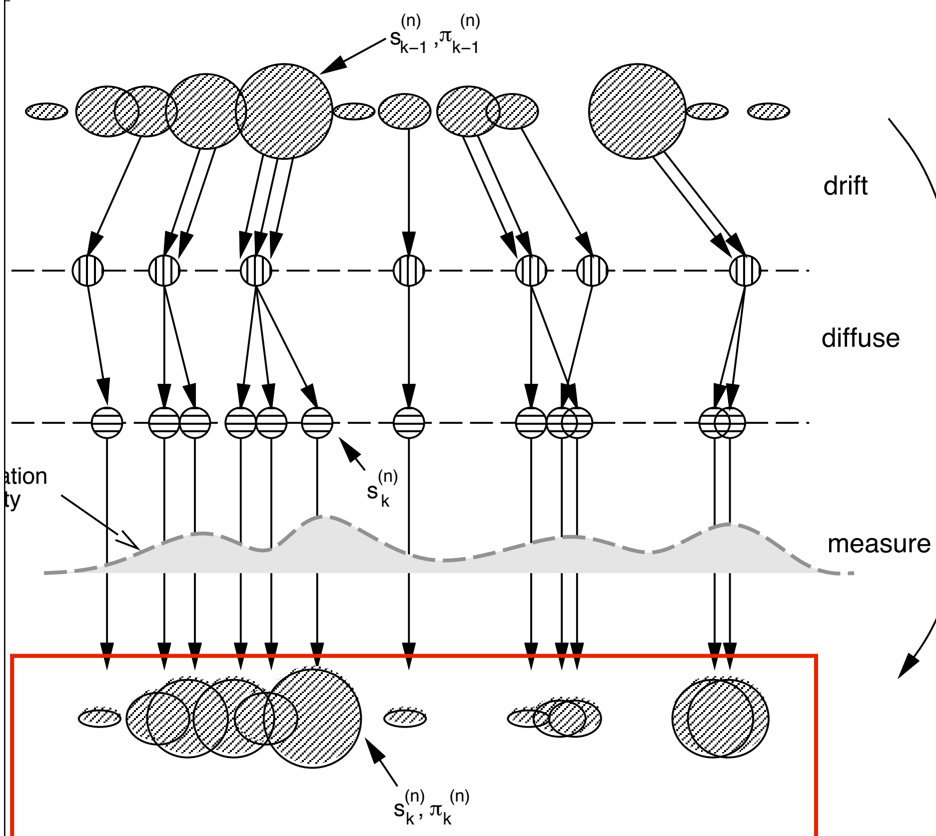
Once the N samples have been constructed: **estimate**, if desired, moments of the tracked position at time-step t as

$$\mathcal{E}[f(\mathbf{x}_t)] = \sum_{n=1}^{N} \pi_t^{(n)} f\left(\mathbf{s}_t^{(n)}\right)$$

obtaining, for instance, a mean position using $f(\mathbf{x}) = \mathbf{x}$.

Figure 6: **The CONDENSATION algorithm.**

estimate state as the mean (weighted average) of particles



Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \ldots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s}_t'^{(n)}$ as follows:
   (a) generate a random number $r \in [0, 1]$, uniformly distributed.
   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$
   (c) set $\mathbf{s}_t'^{(n)} = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from
$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s}_t'^{(n)})$$
   to choose each $\mathbf{s}_t^{(n)}$. For instance, in the case that the dynamics are governed by a linear stochastic differential equation, the new sample value may be generated as: $\mathbf{s}_t^{(n)} = \mathbf{A}\mathbf{s}_t'^{(n)} + B\mathbf{w}_t^{(n)}$ where $\mathbf{w}_t^{(n)}$ is a vector of standard normal random variates, and BB^T is the process noise covariance — see section 5.
3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:
$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$
   then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where
$$\begin{aligned} c_t^{(0)} &= 0, \\ c_t^{(n)} &= c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1, \ldots, N). \end{aligned}$$

Once the N samples have been constructed: **estimate**, if desired, moments of the tracked position at time-step t as

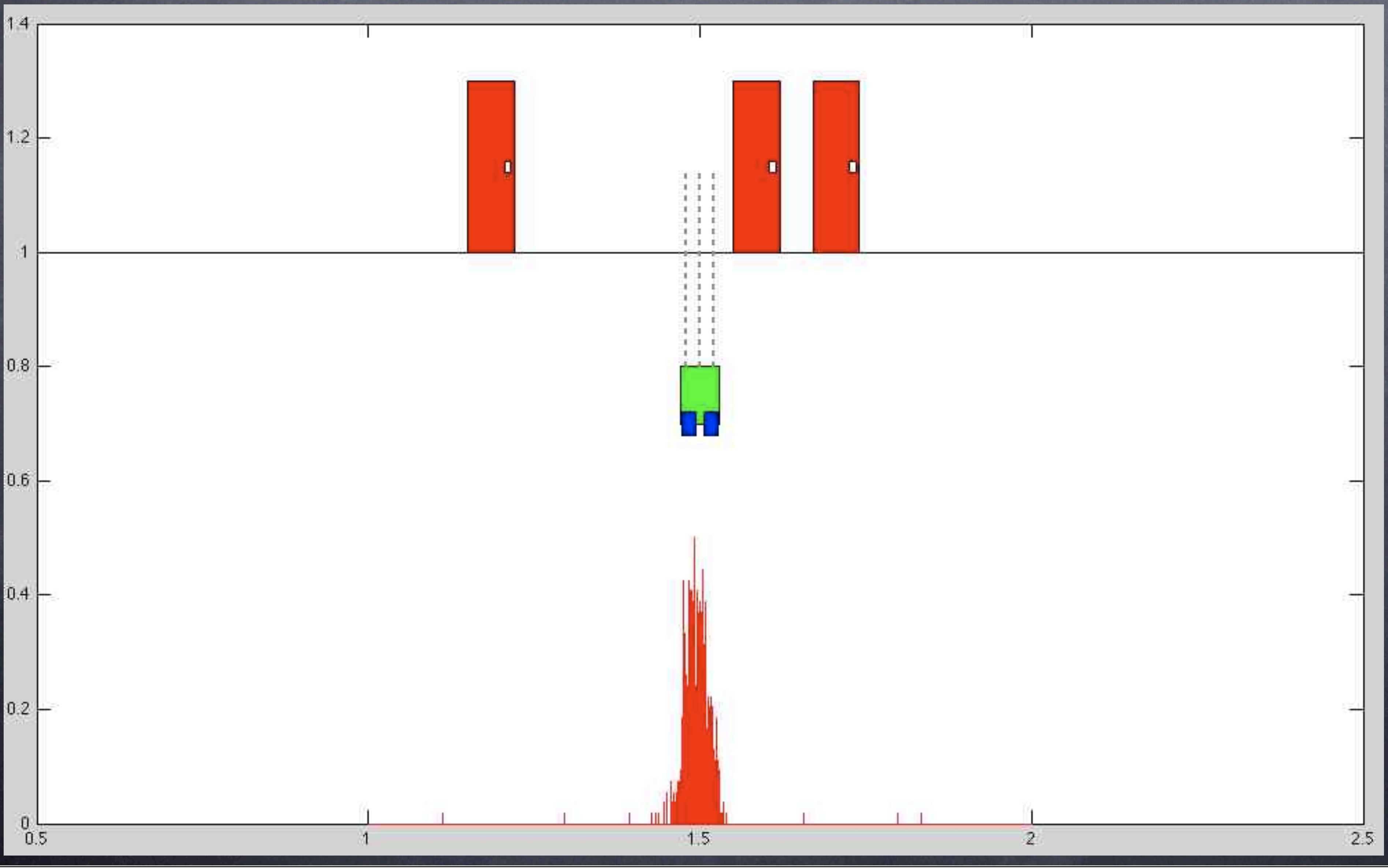
$$\mathcal{E}[f(\mathbf{x}_t)] = \sum_{n=1}^{N} \pi_t^{(n)} f\left(\mathbf{s}_t^{(n)}\right)$$

obtaining, for instance, a mean position using $f(\mathbf{x}) = \mathbf{x}$.

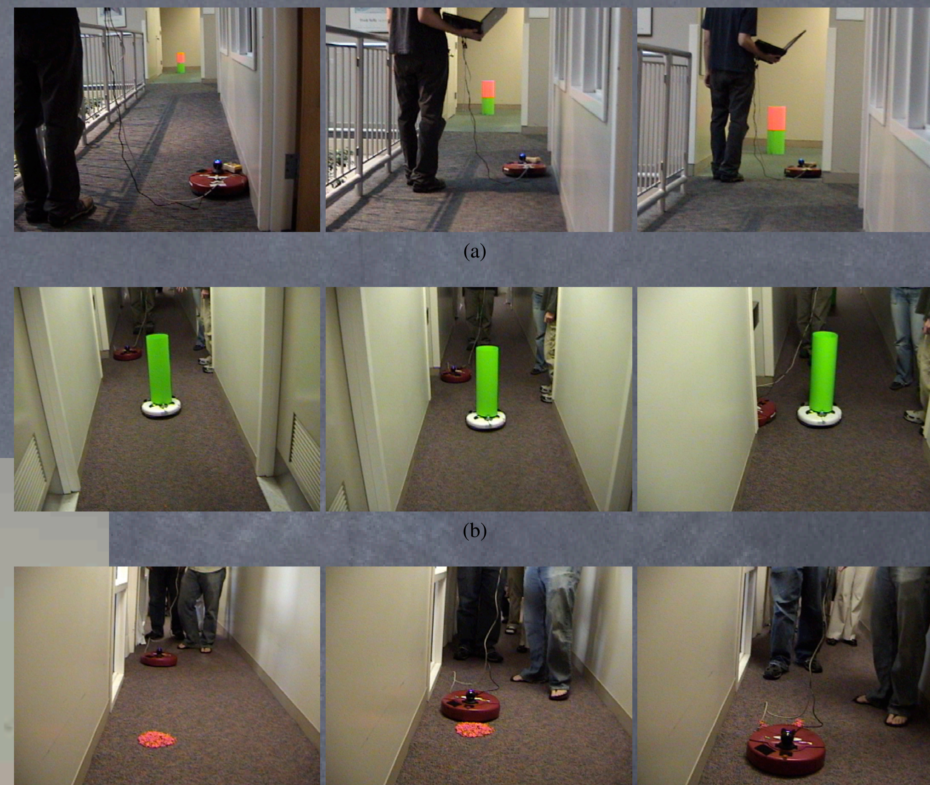
Figure 6: **The CONDENSATION algorithm**

matlab example

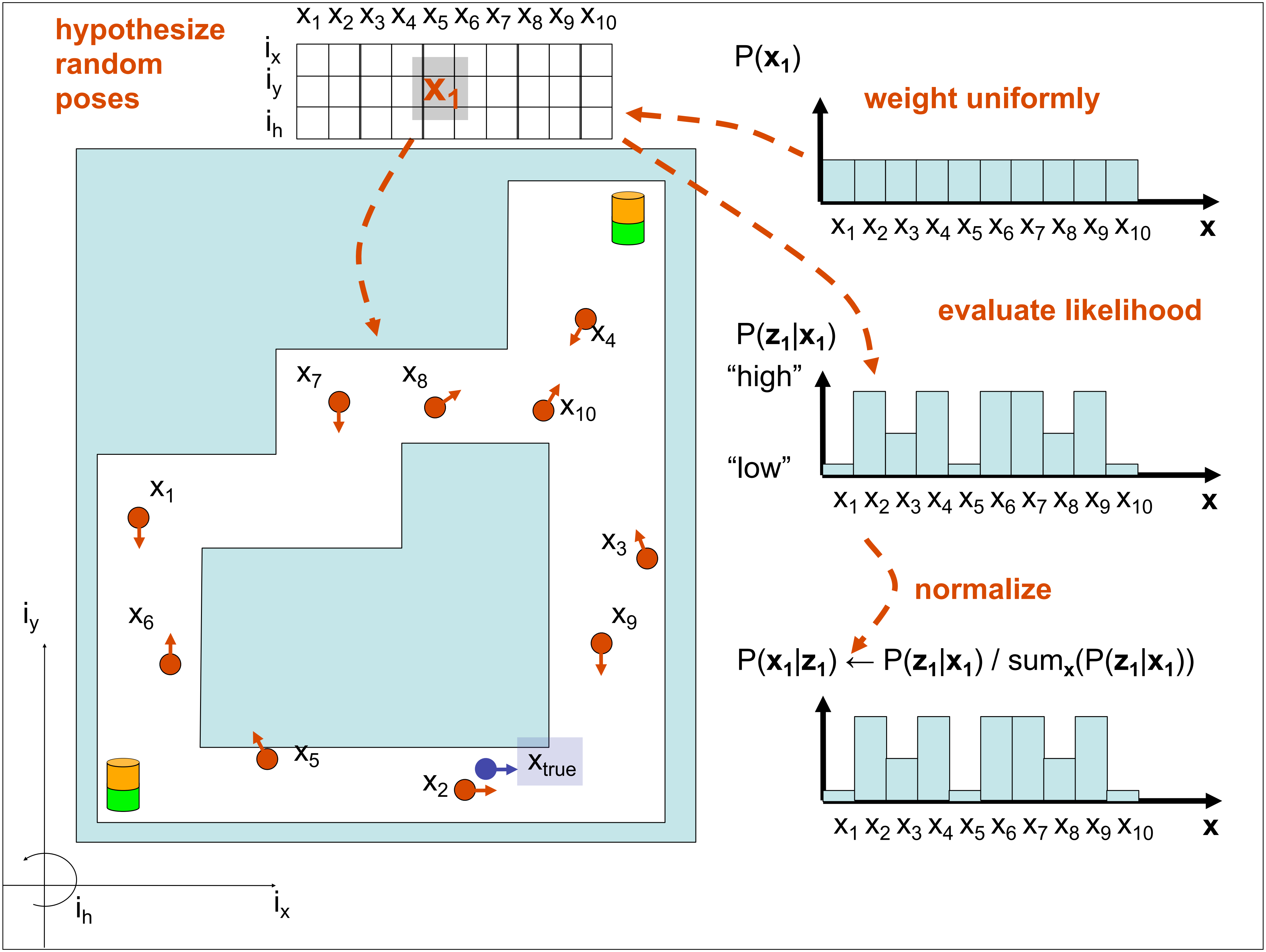
- /course/cs148/pub/particlefilter_hallway.m



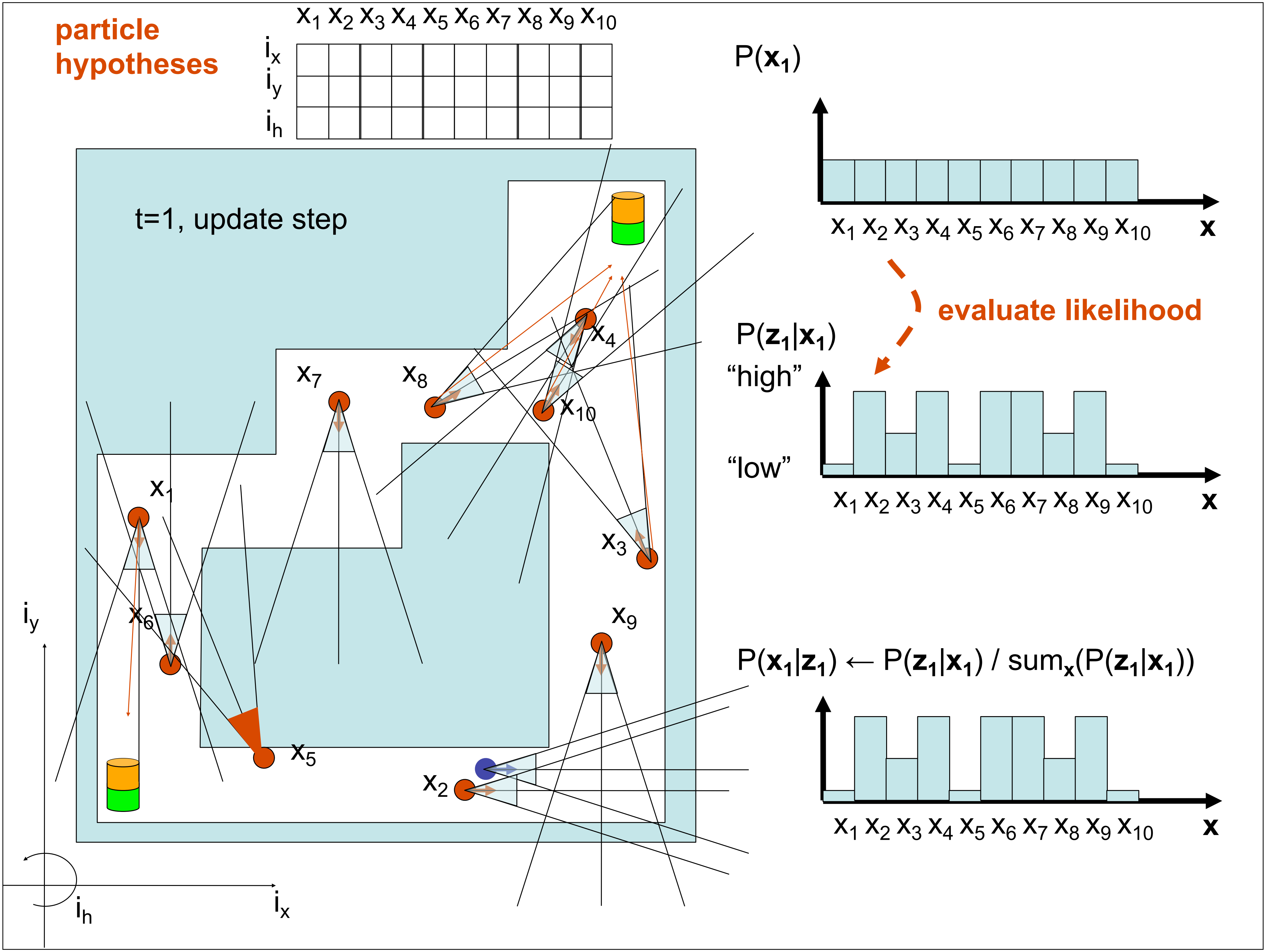
CIT 5th floor localization example (Roomba Pac-Man)



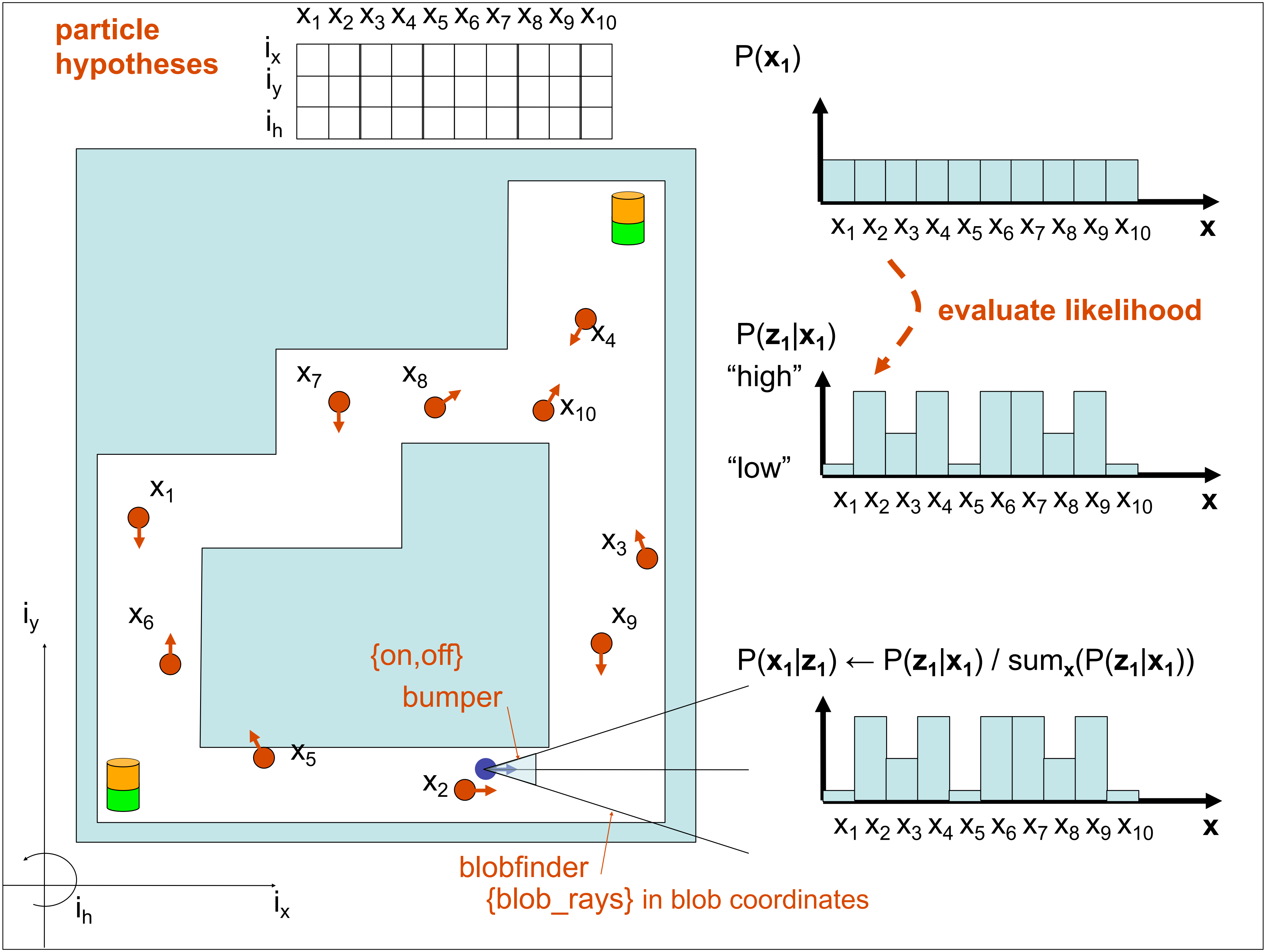
(a) (b) (c)



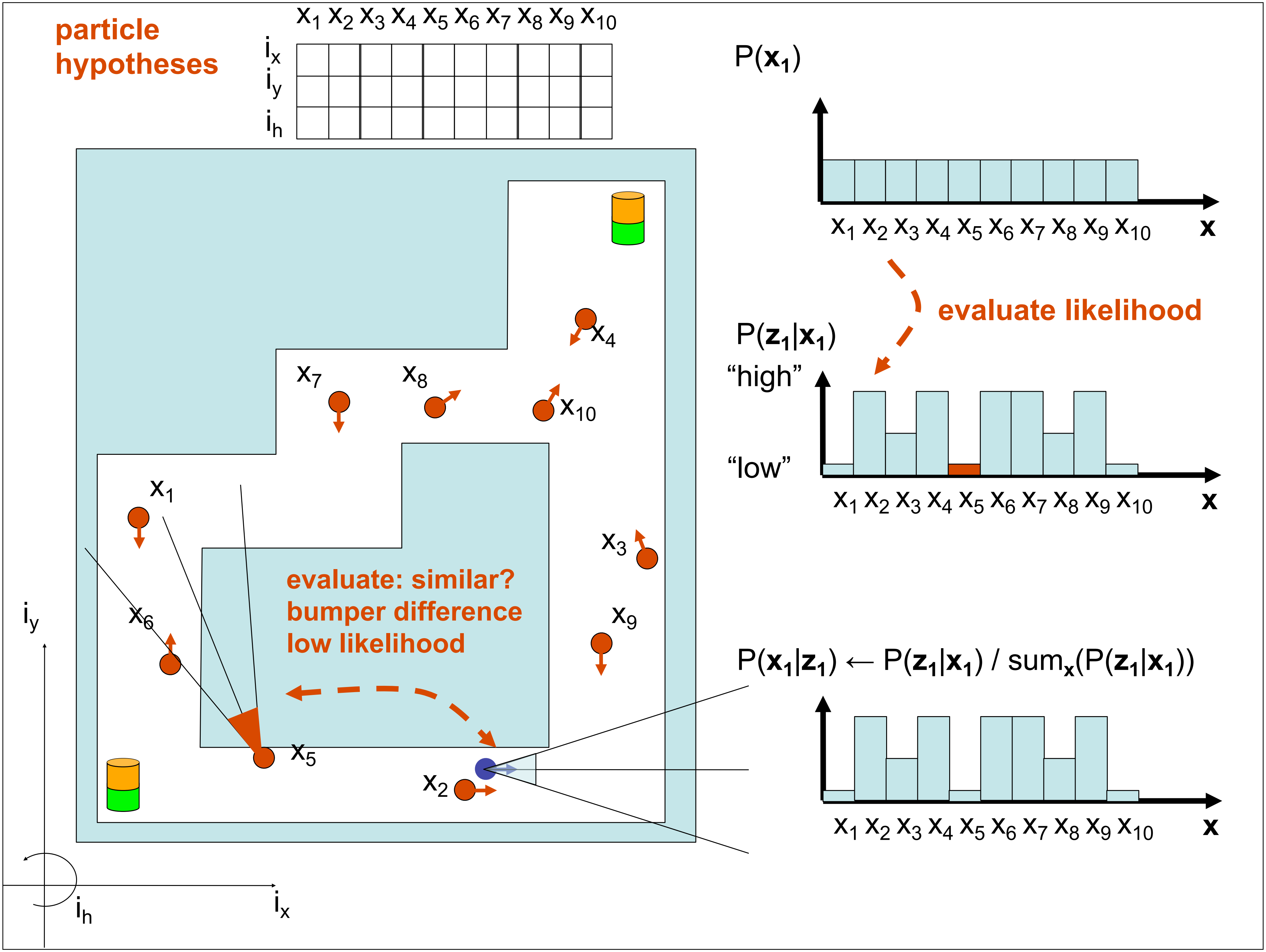
hypothesize random poses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x1
P(x1)
weight uniformly
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
evaluate likelihood
P(z1|x1)
"high"
"low"
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
normalize
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
x4
x7
x8
x10
x1
x3
x6
x9
x5
xtrue
x2
iy
ih
ix



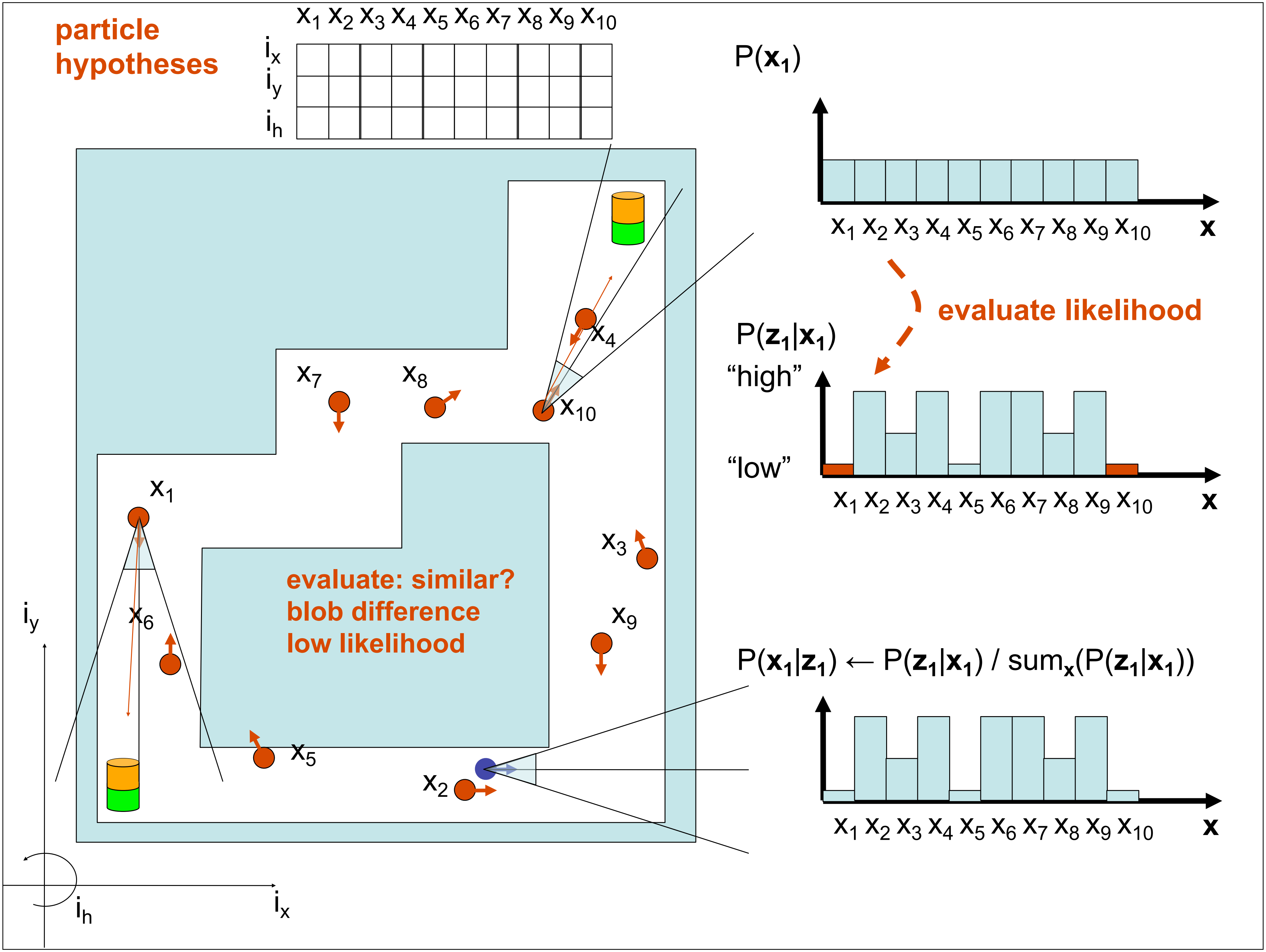
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
t=1, update step
x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
iy
ih
ix
P(x1)
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
evaluate likelihood
P(z1|x1)
"high"
"low"
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x



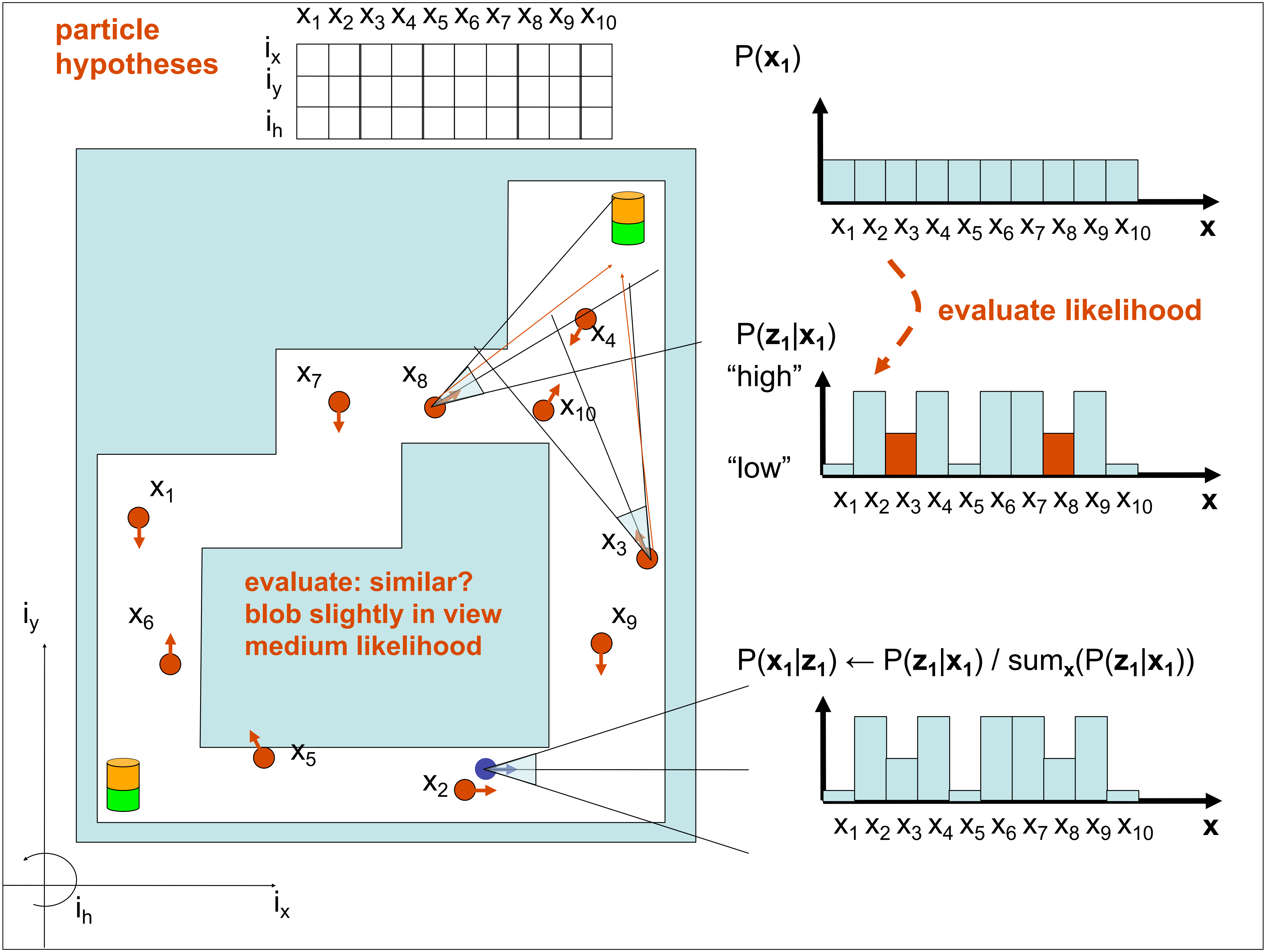
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
P(x1)
evaluate likelihood
P(z1|x1)
"high"
"low"
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
{on,off}
bumper
blobfinder
{blob_rays} in blob coordinates



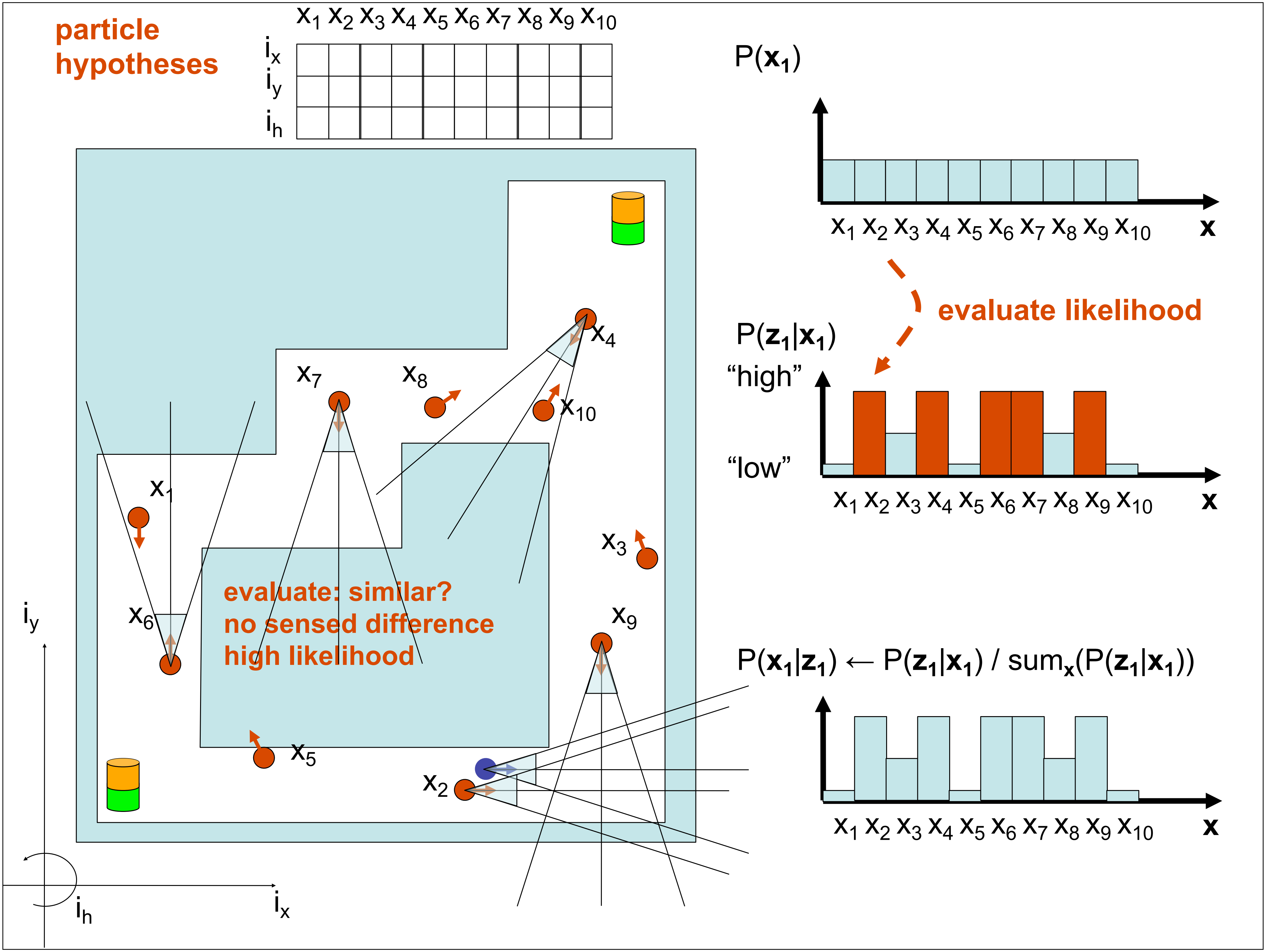
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
P(x1)
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
evaluate likelihood
P(z1|x1)
“high”
“low”
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
x4
x7
x8
x10
x1
x3
evaluate: similar?
bumper difference
low likelihood
x6
x9
x5
x2
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
iy
ih
ix



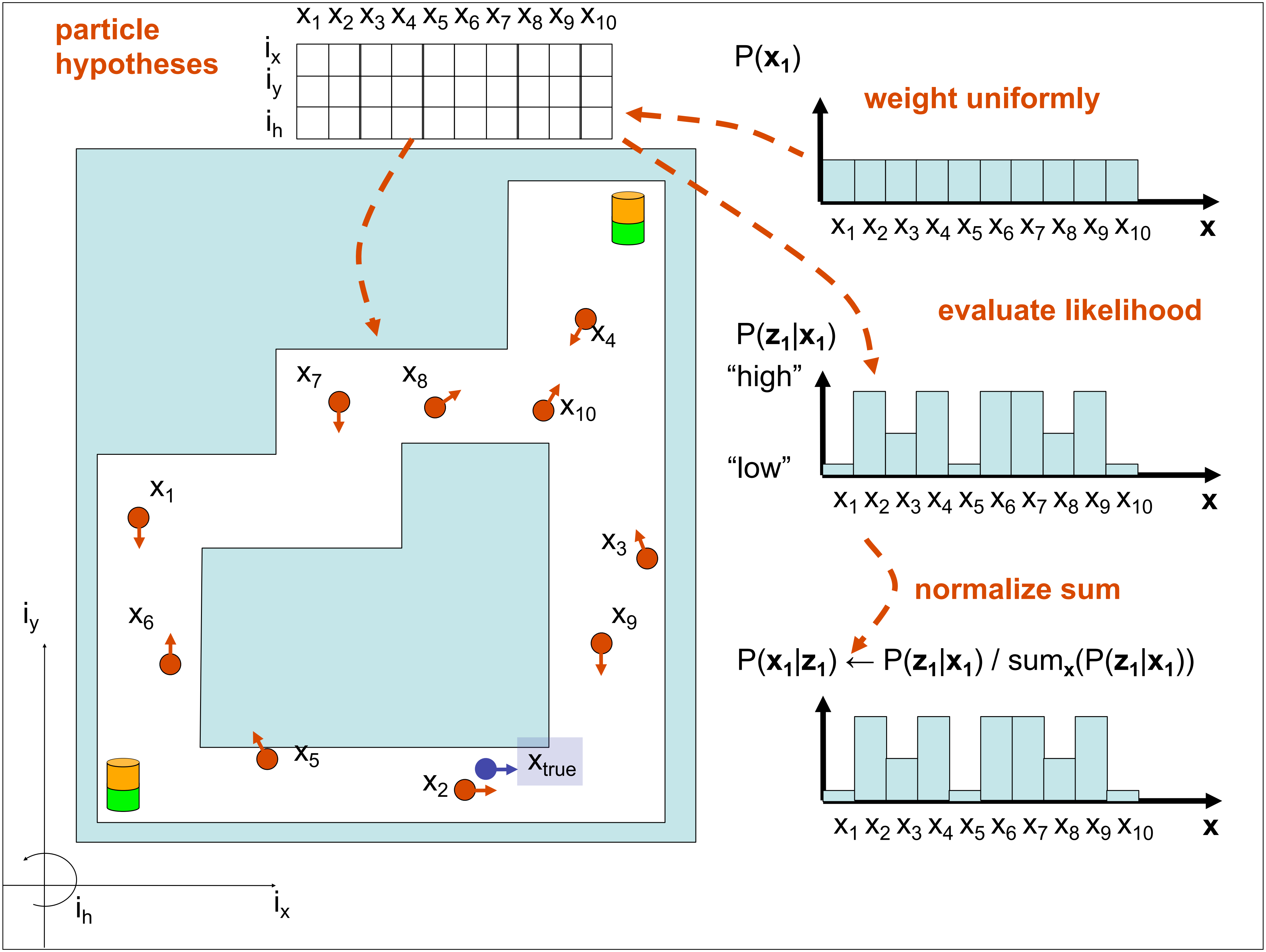
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
P(x1)
evaluate likelihood
P(z1|x1)
"high"
"low"
evaluate: similar?
blob difference
low likelihood
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
x



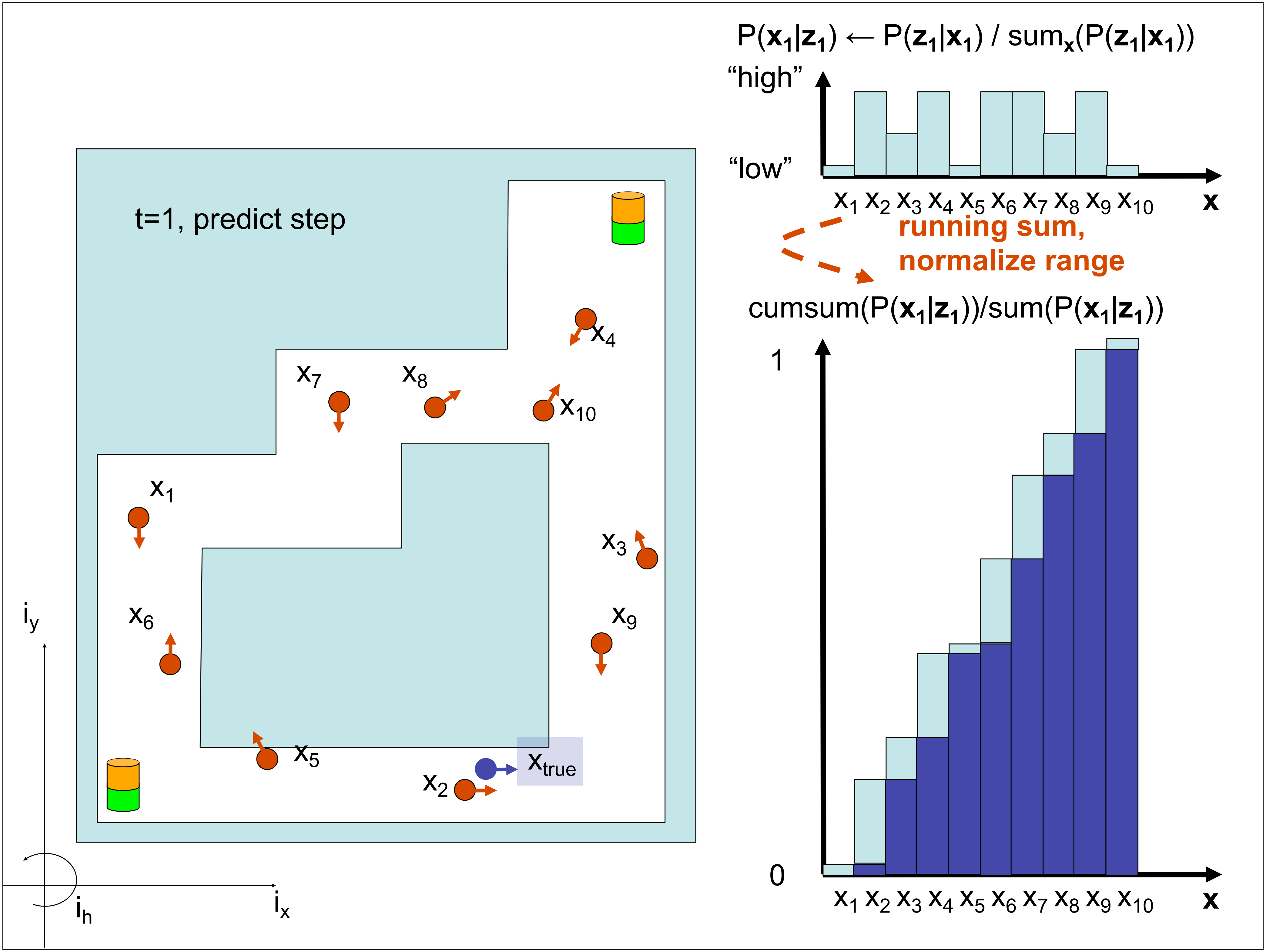
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
P(x1)
x
evaluate likelihood
P(z1|x1)
"high"
"low"
x7
x8
x4
x10
x1
x3
evaluate: similar?
blob slightly in view
medium likelihood
x6
x9
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
x5
x2
iy
ih
ix



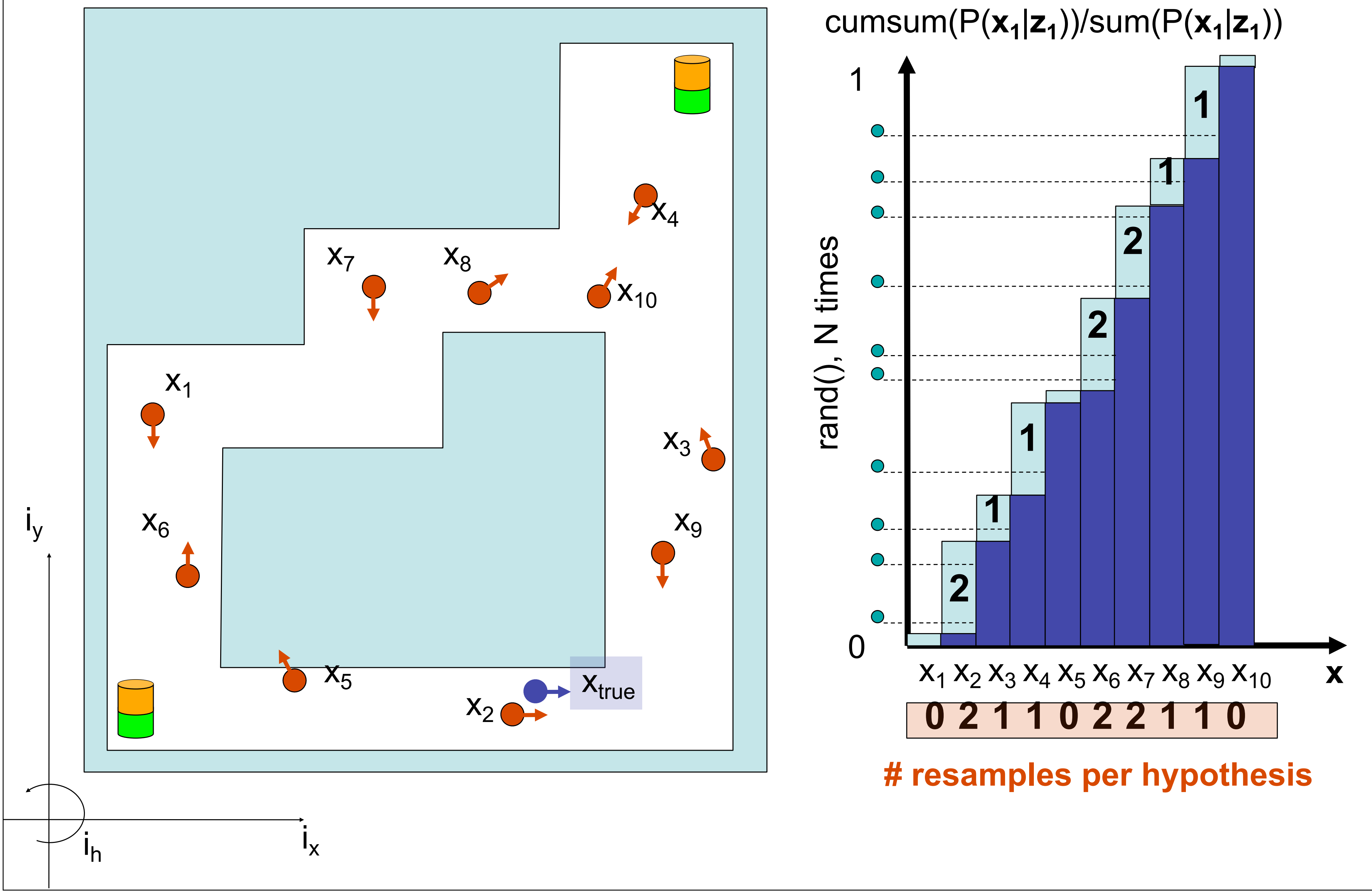
particle hypotheses
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10}
i_x
i_y
i_h
$P(\mathbf{x_1})$
x
evaluate likelihood
$P(\mathbf{z_1}|\mathbf{x_1})$
“high”
“low”
evaluate: similar?
no sensed difference
high likelihood
$P(\mathbf{x_1}|\mathbf{z_1}) \leftarrow P(\mathbf{z_1}|\mathbf{x_1}) / \mathrm{sum}_\mathbf{x}(P(\mathbf{z_1}|\mathbf{x_1}))$



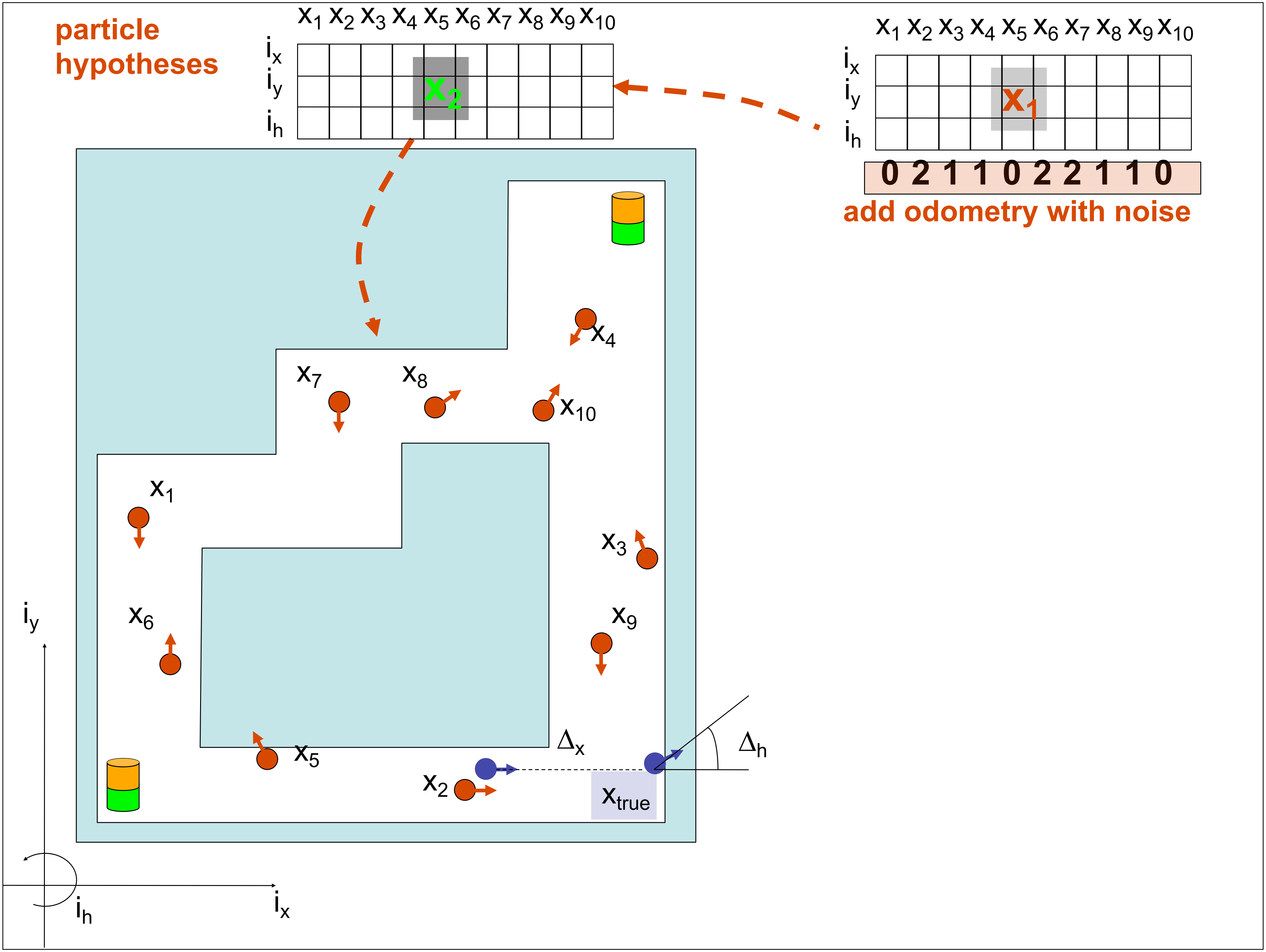
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
P(x1)
weight uniformly
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
evaluate likelihood
P(z1|x1)
“high”
“low”
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
normalize sum
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10 x
x4
x7
x8
x10
x1
x3
x6
x9
x5
xtrue
x2
iy
ih
ix

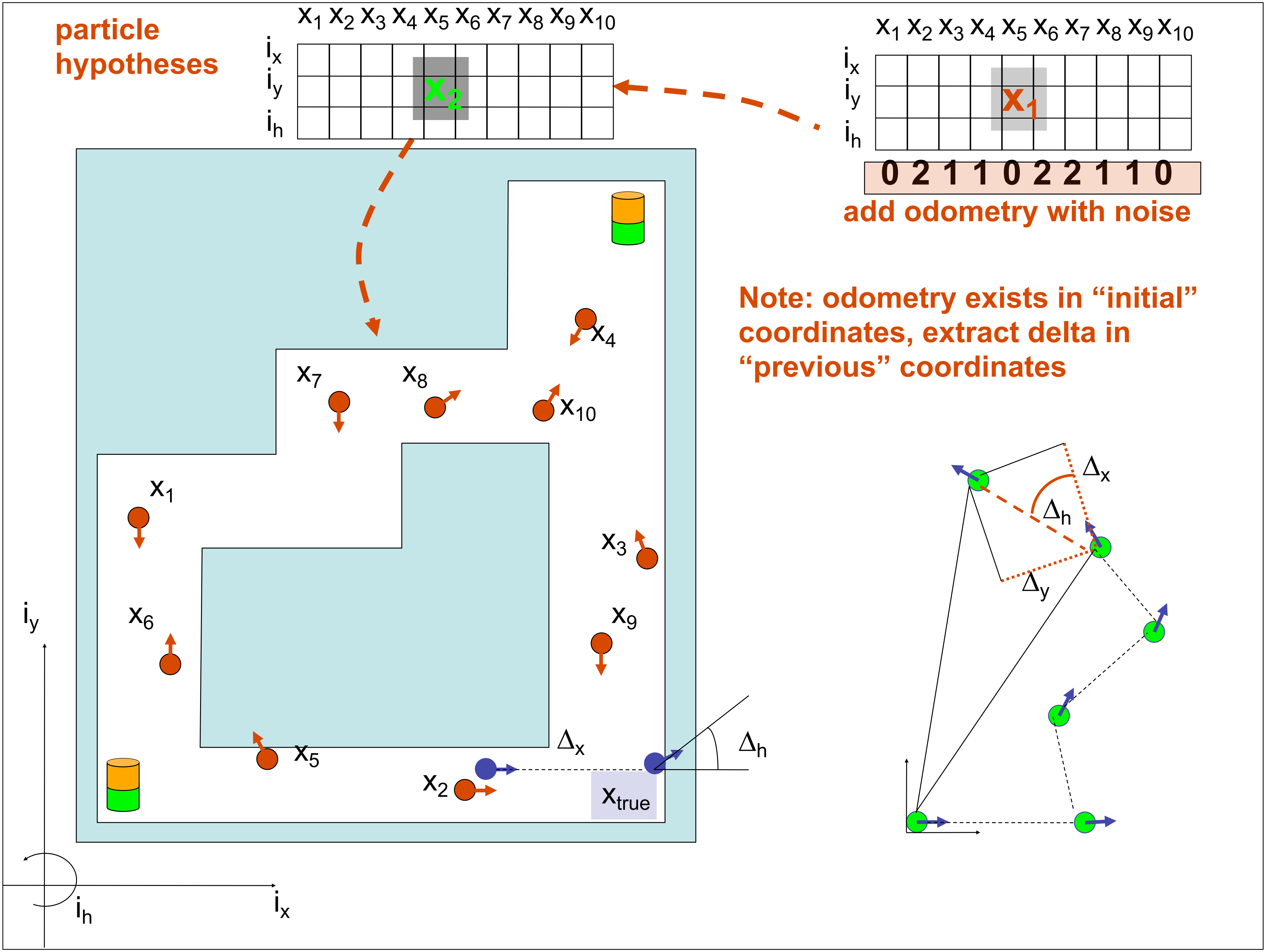


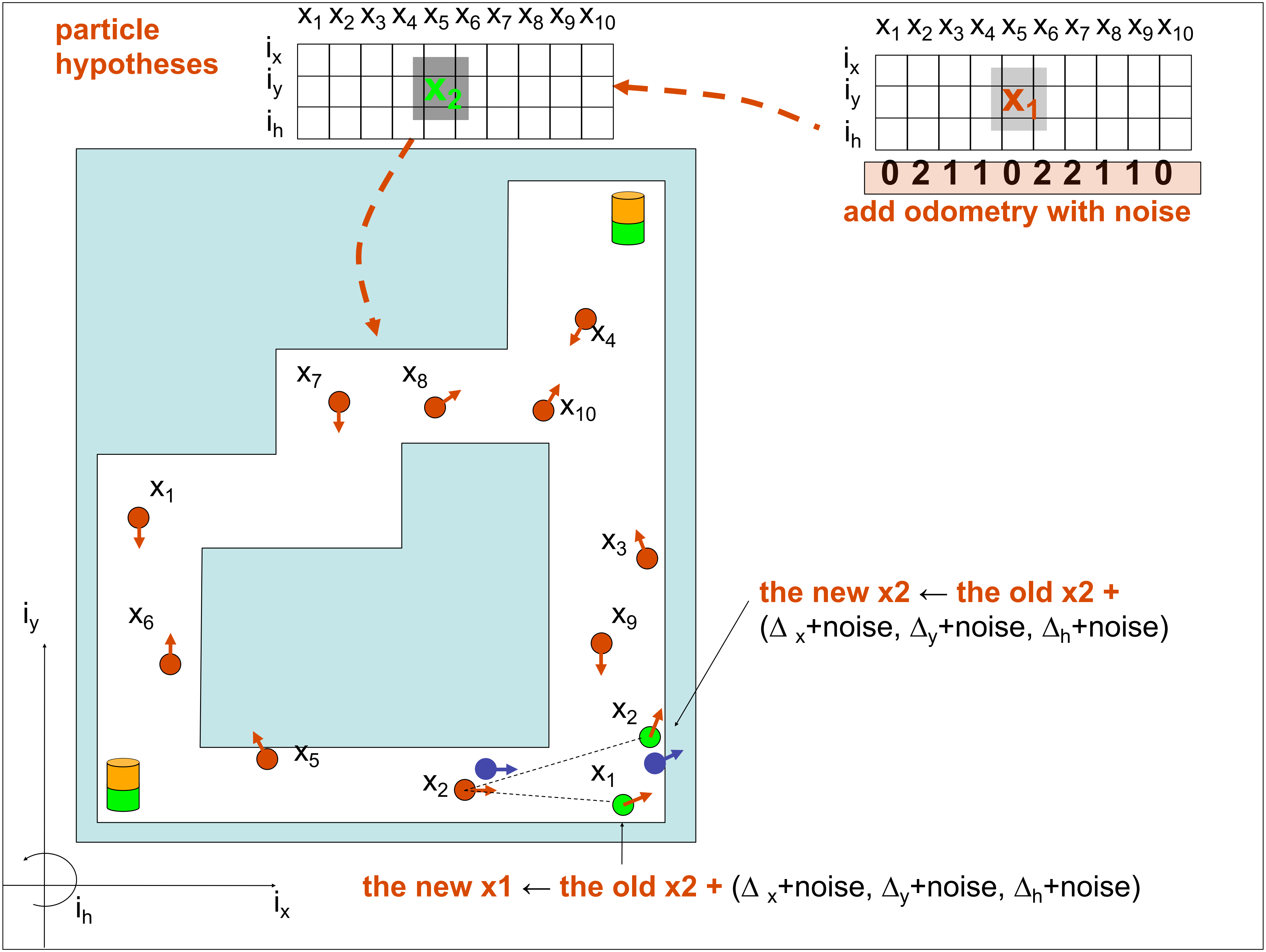
P(x1|z1) ← P(z1|x1) / sumx(P(z1|x1))
“high”
“low”
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
running sum,
normalize range
cumsum(P(x1|z1))/sum(P(x1|z1))
1
0
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
t=1, predict step
x4
x7
x8
x10
x1
x3
x6
x9
x5
xtrue
x2
iy
ih
ix

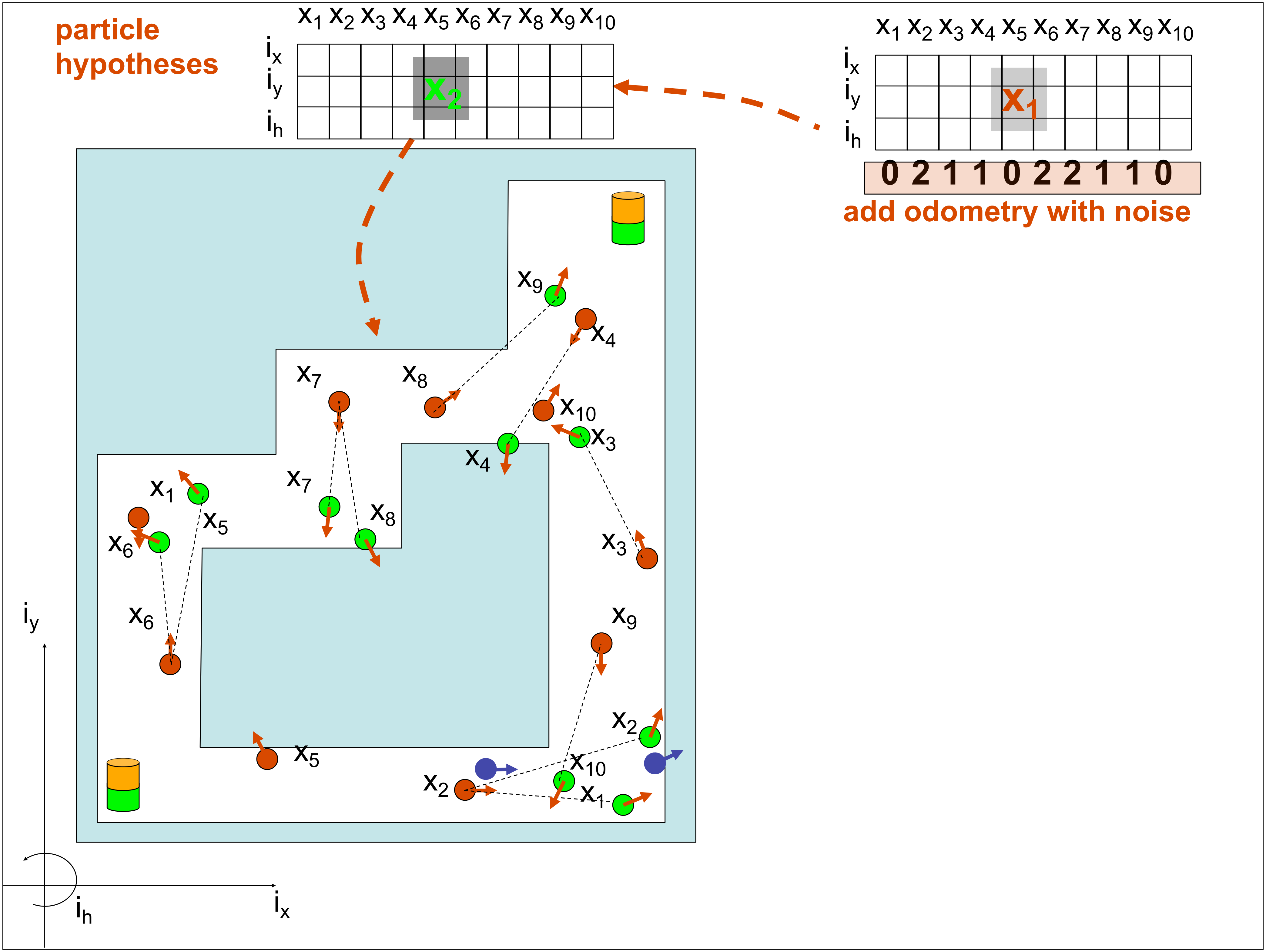


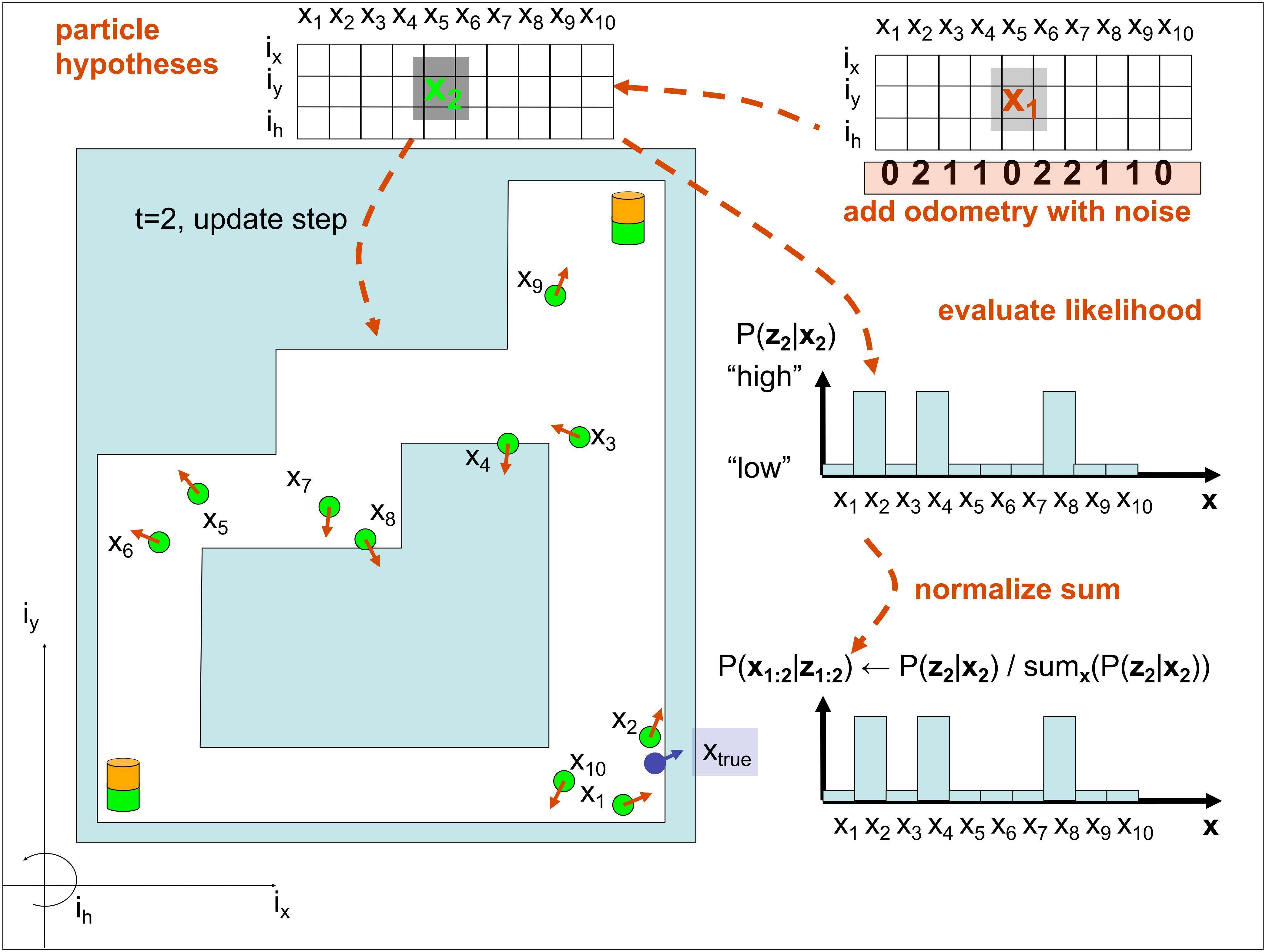
x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
xtrue
iy
ix
ih
cumsum(P(x1|z1))/sum(P(x1|z1))
1
0
rand(), N times
2
1
1
2
2
1
1
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
0 2 1 1 0 2 2 1 1 0
resamples per hypothesis


particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x2
x1
0 2 1 1 0 2 2 1 1 0
add odometry with noise
x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
Δx
Δh
xtrue
iy
ix
ih

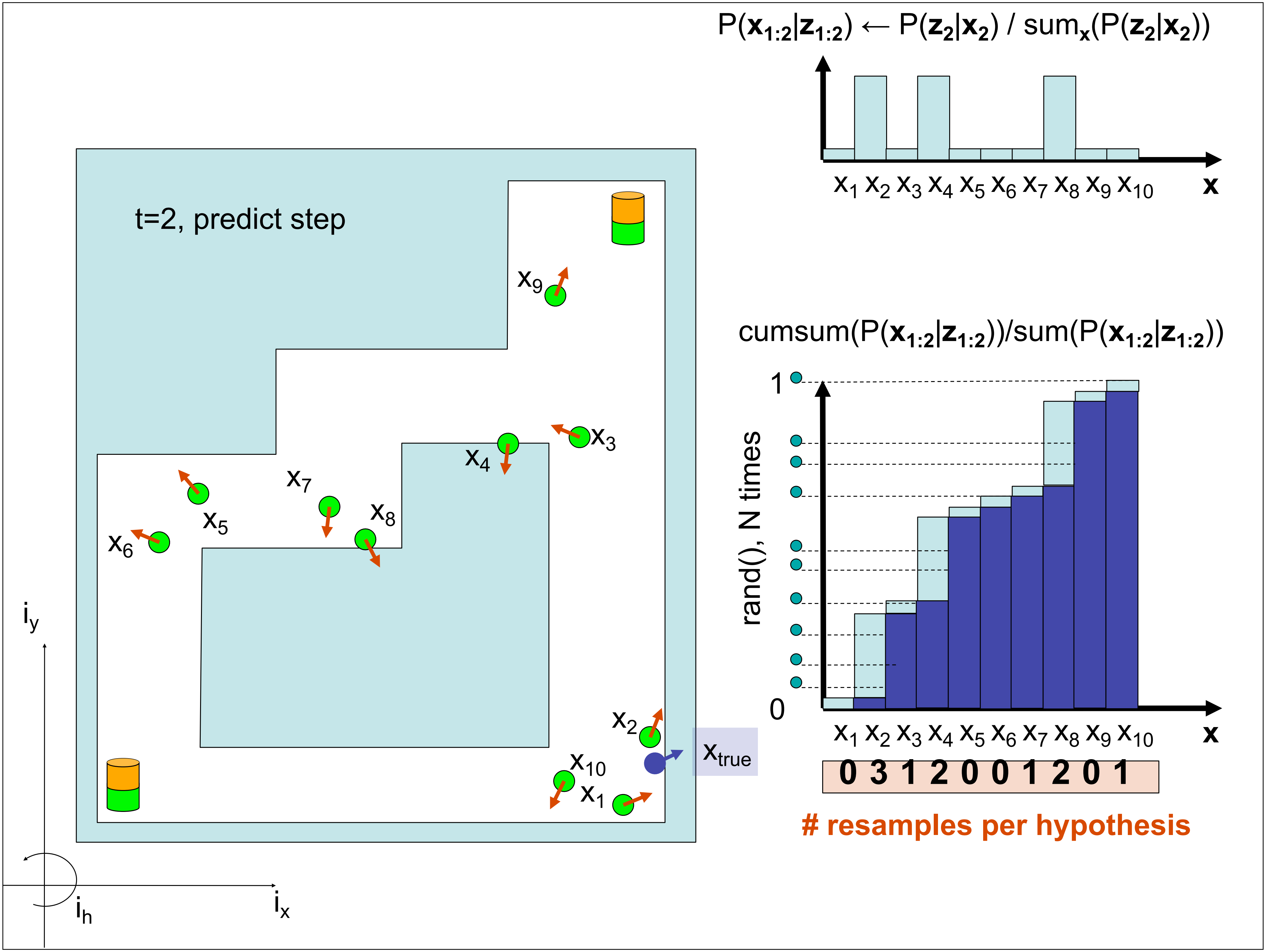

particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x2
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x1
0 2 1 1 0 2 2 1 1 0
add odometry with noise
Note: odometry exists in “initial” coordinates, extract delta in “previous” coordinates
x4
x7
x8
x10
x1
x3
x6
x9
x5
x2
xtrue
Δx
Δh
Δy
iy
ih
ix


particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x2
x1
0 2 1 1 0 2 2 1 1 0
add odometry with noise
x4
x7
x8
x10
x1
x3
x6
x9
x2
x5
x2
x1
iy
ih
ix
the new x2 ← the old x2 +
(Δ x+noise, Δy+noise, Δh+noise)
the new x1 ← the old x2 + (Δ x+noise, Δy+noise, Δh+noise)

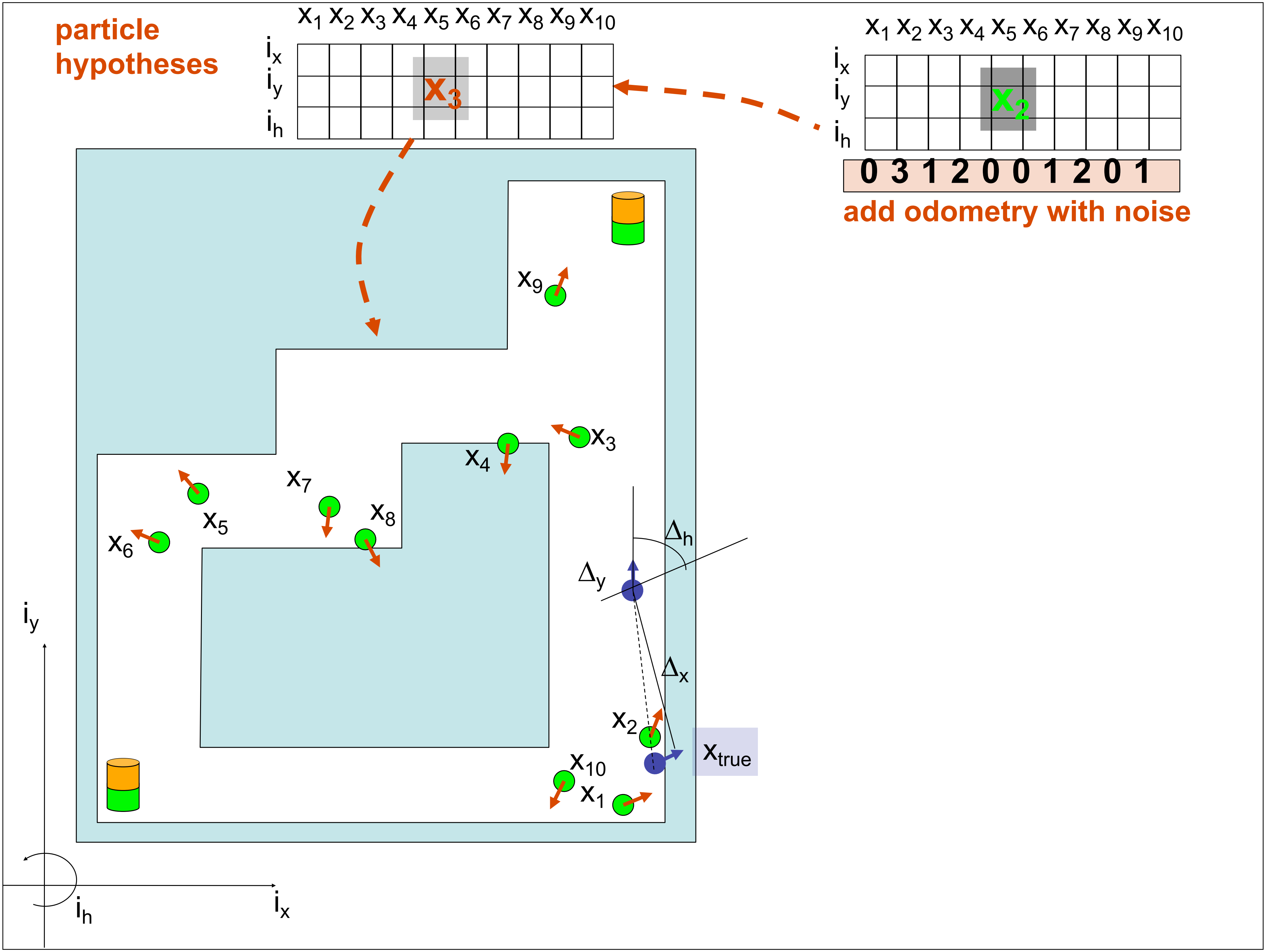

particle hypotheses
add odometry with noise
0 2 1 1 0 2 2 1 1 0
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih



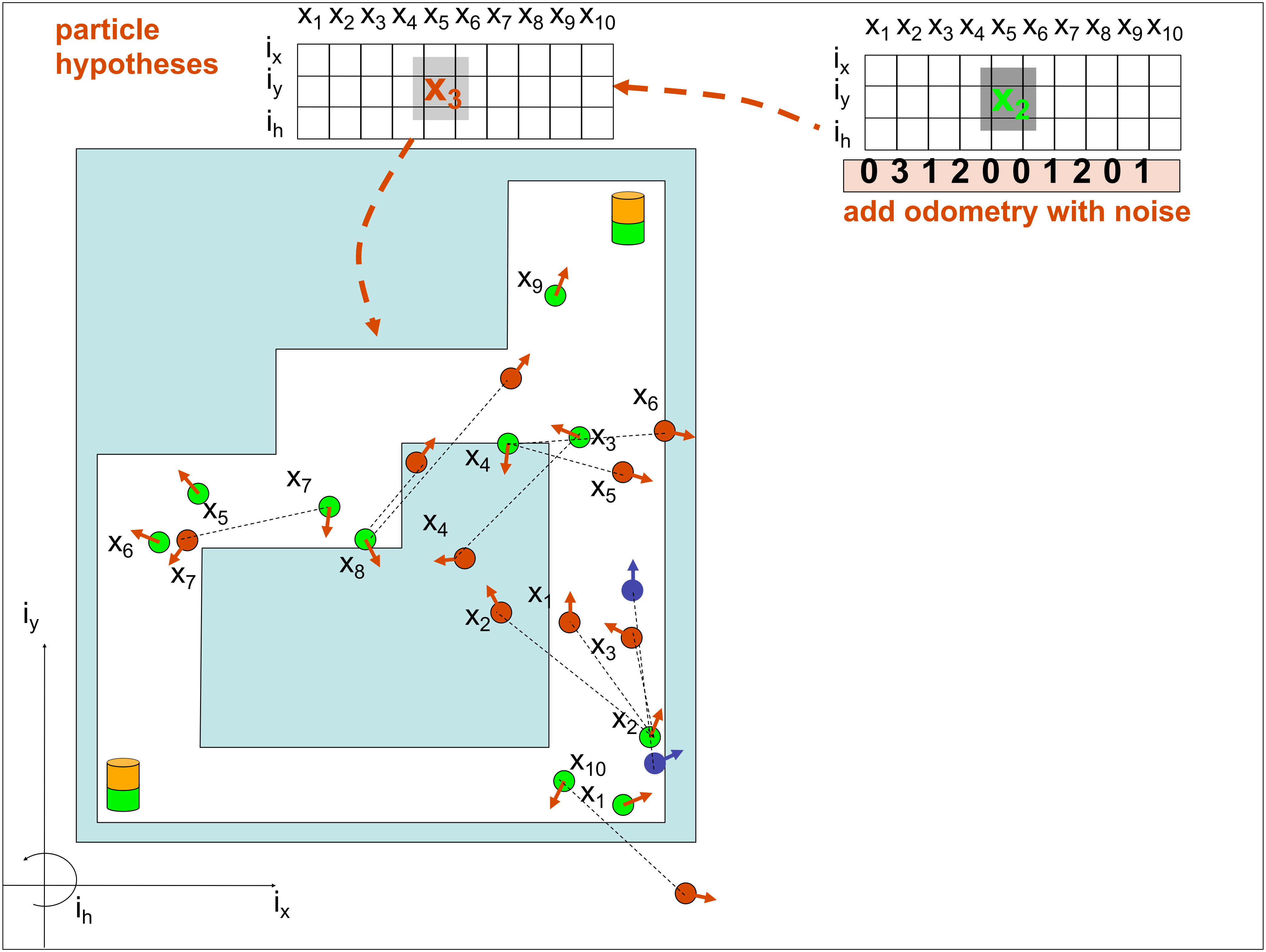
particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x2
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x1
0 2 1 1 0 2 2 1 1 0
add odometry with noise
t=2, update step
x9
x3
x4
x7
x8
x5
x6
x2
xtrue
x10
x1
evaluate likelihood
P(z2|x2)
"high"
"low"
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
normalize sum
P(x1:2|z1:2) ← P(z2|x2) / sumx(P(z2|x2))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
iy
ih
ix

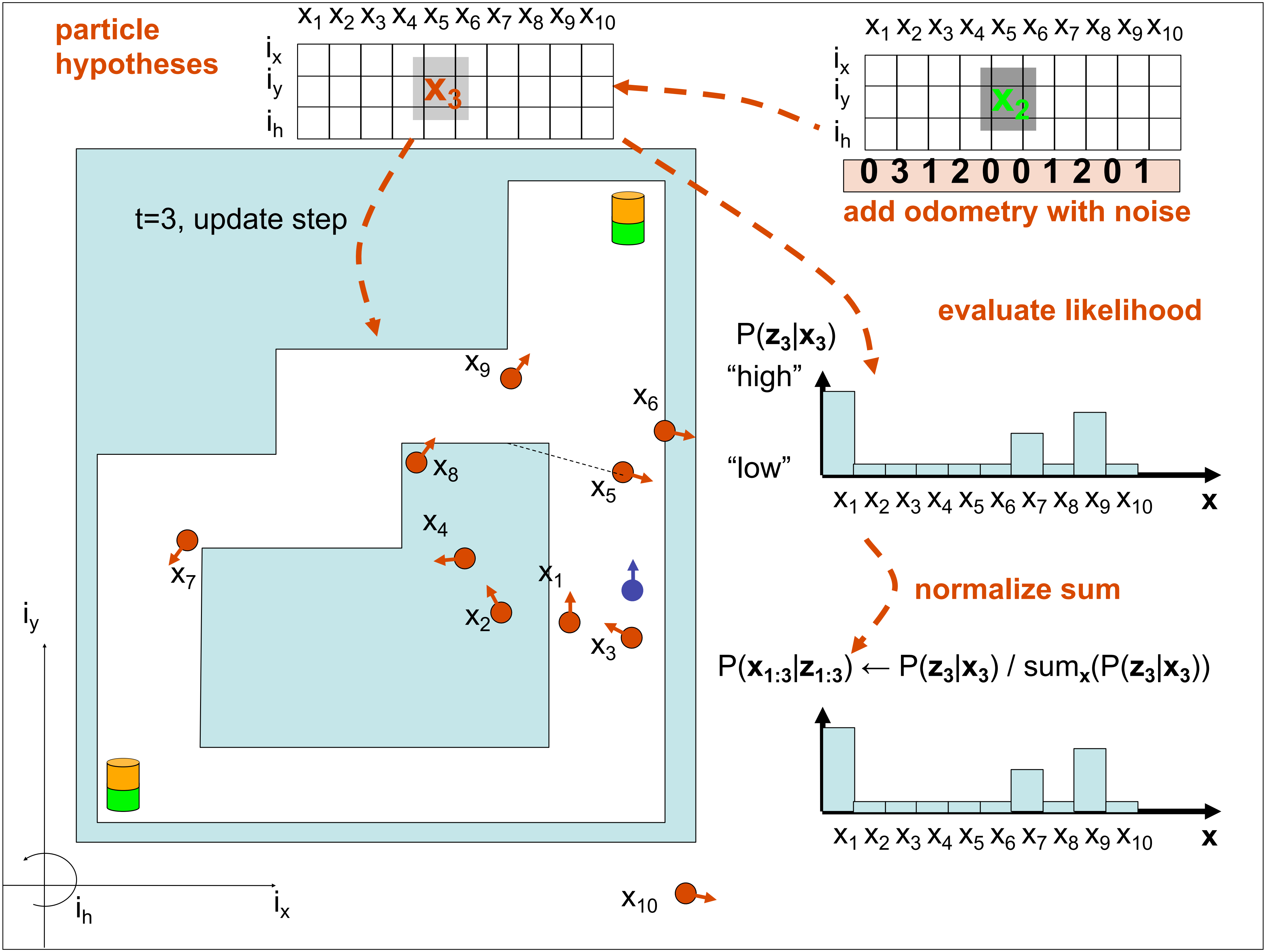


P(x1:2|z1:2) ← P(z2|x2) / sumx(P(z2|x2))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
t=2, predict step
x9
x3
x4
x7
x5
x8
x6
iy
x2
x10
xtrue
x1
ih
ix
cumsum(P(x1:2|z1:2))/sum(P(x1:2|z1:2))
1
0
rand(), N times
0 3 1 2 0 0 1 2 0 1
resamples per hypothesis

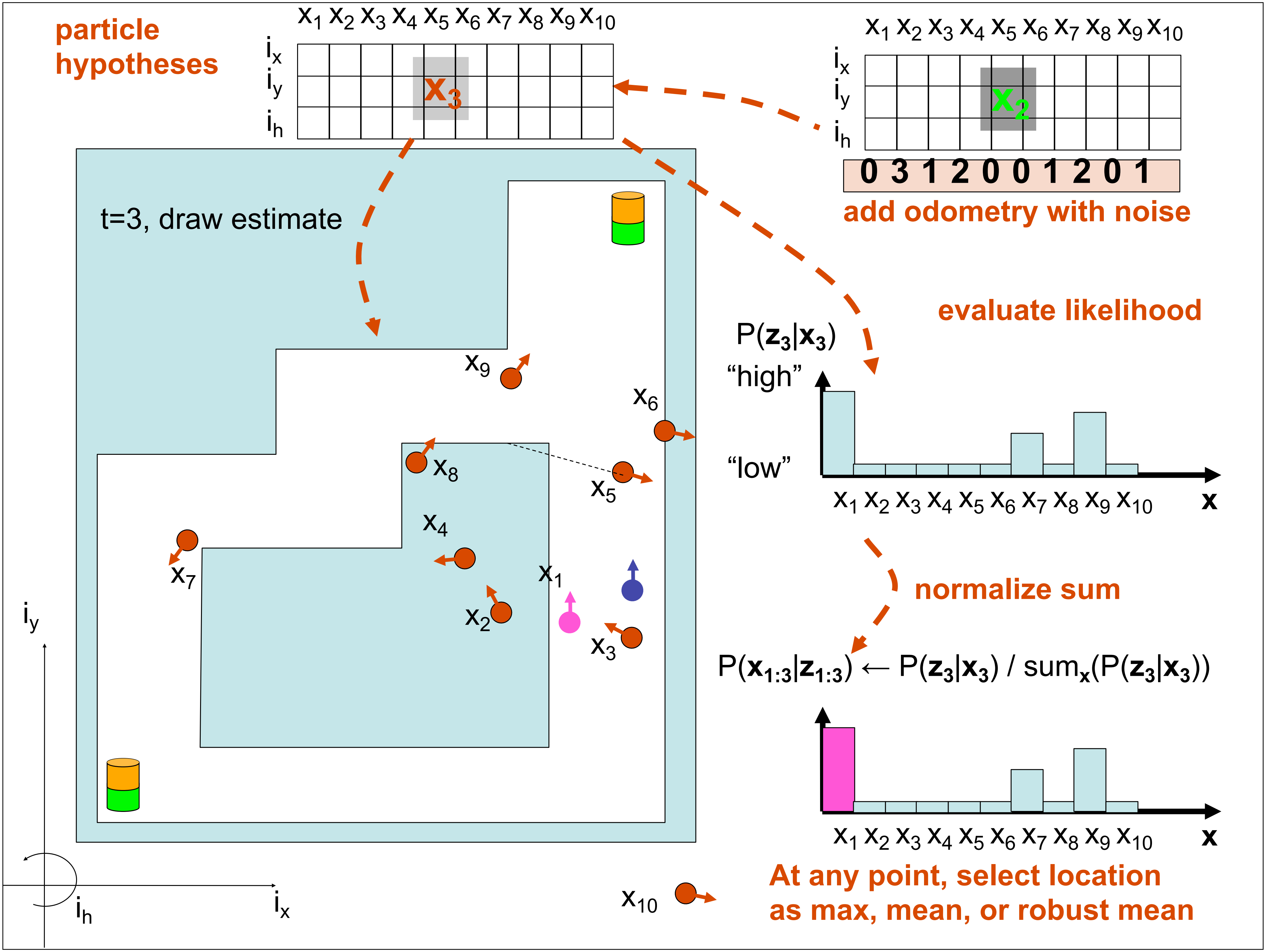


particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x3
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x2
0 3 1 2 0 0 1 2 0 1
add odometry with noise
x9
x3
x4
x7
x5
x8
x6
Δh
Δy
Δx
x2
x10
x1
xtrue
iy
ih
ix


particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x3
x2
0 3 1 2 0 0 1 2 0 1
add odometry with noise
x9
x6
x3
x4
x5
x7
x5
x6
x7
x8
x4
x1
x2
x3
x2
x10
x1
iy
ih
ix



particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x3
x2
0 3 1 2 0 0 1 2 0 1
add odometry with noise
t=3, update step
evaluate likelihood
P(z3|x3)
"high"
"low"
x9
x6
x8
x5
x4
x7
x1
x2
x3
x10
x
normalize sum
P(x1:3|z1:3) ← P(z3|x3) / sumx(P(z3|x3))



particle hypotheses
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
ix
iy
ih
x3
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x2
0 3 1 2 0 0 1 2 0 1
add odometry with noise
t=3, draw estimate
evaluate likelihood
P(z3|x3)
"high"
"low"
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
x9
x6
x8
x5
x4
x7
x1
x2
x3
x10
normalize sum
P(x1:3|z1:3) ← P(z3|x3) / sumx(P(z3|x3))
x1 x2 x3 x4 x5 x6 x7 x8 x9 x10
x
At any point, select location as max, mean, or robust mean
iy
ih
ix

Open Issues

- How many particles?
- How to evaluate likelihood?
- How much odometry noise?
- What if my estimate is wrong?